

Bergische Universität - GH Wuppertal
Lehr- und Forschungsgebiet Bauinformatik

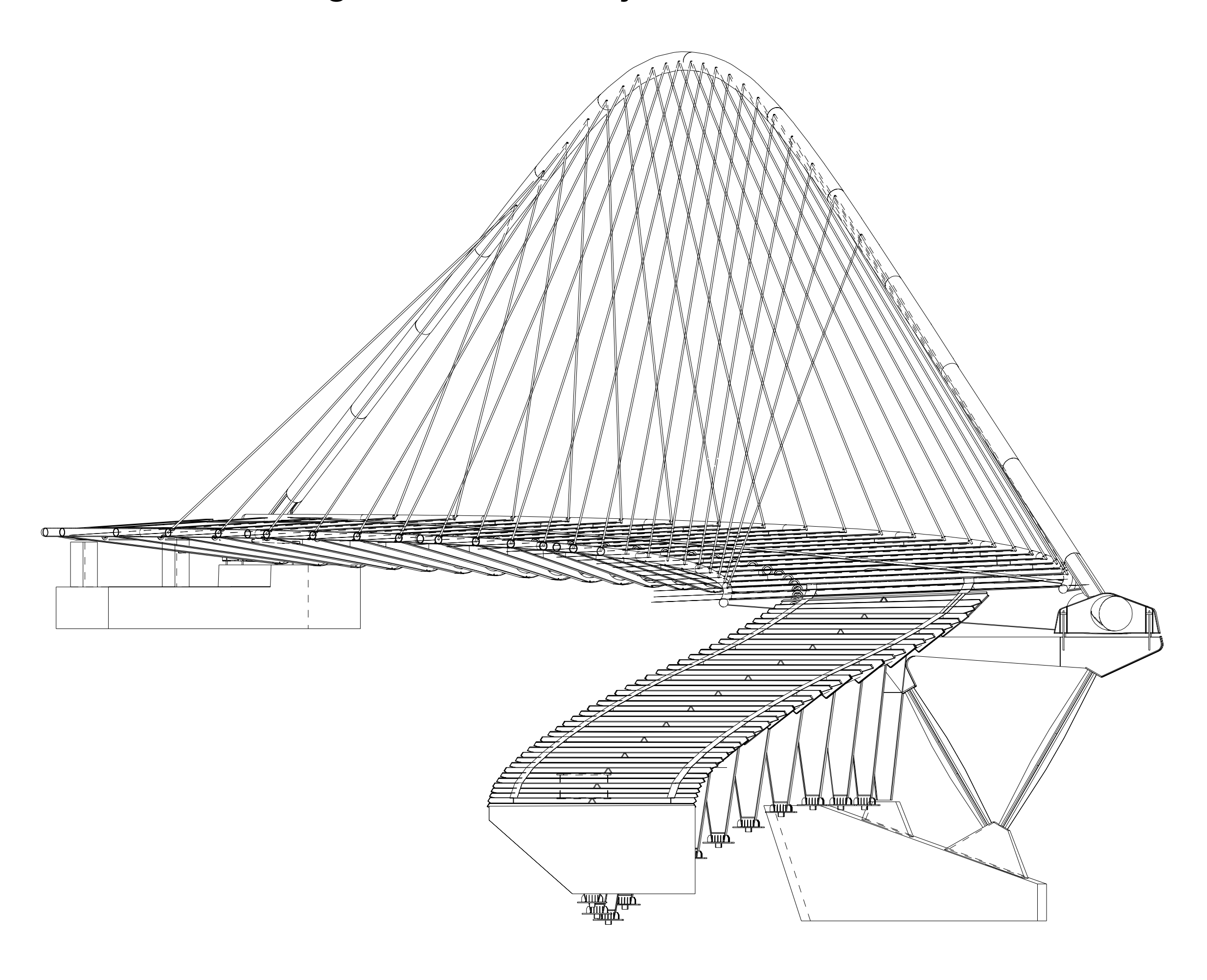
Angewandte Informatik und Theoretische Methoden
im Bauingenieurwesen

Yongyu Chang

Eine von Landessprachen unabhängige Nutzoberfläche mit intelligenten CAD-Objekten des Bauwesens

Eine von Landessprachen unabhängige Nutzoberfläche

mit intelligenten CAD-Objekten des Bauwesens



Dem Fachbereich Bauingenieurwesen der Bergischen Universität-
Gesamthochschule Wuppertal vorgelegte
Dissertation
zur Erlangung des Grades Doktor-Ingenieur

von
Yongyu Chang
VR China

Wuppertal 2002



Bergische Universität - GH Wuppertal
Lehr- und Forschungsgebiet Bauinformatik

Dissertation

Genehmigt vom Fachbereich Bauingenieurwesen
der Bergischen Universität-GH Wuppertal

Dissertationsschrift eingereicht : 08. Juli 2002
Mündliche Prüfung und Disputation: 15. Oktober 2002

Gutachter: Univ.-Prof. Dr.-Ing. G. Pegels
Univ.-Prof. Dr.-Ing. F. Huber

Vorsitzender: Univ.-Prof. Dr.-Ing. C. J. Diederichs

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter im Lehrgebiet für Angewandte Informatik und Theoretische Methoden im Bauingenieurwesen, Fachbereich 11, an der Bergischen Universität-Gesamthochschule Wuppertal.

Mein besonderer Dank gilt Herrn Univ.-Prof. Dr.-Ing. G. Pegels für Anregungen und Unterstützung dieser Arbeit.

Herrn Univ.-Prof. Dr.-Ing. C. J. Diederichs danke ich herzlich für die Übernahme des Korreferats.

Weiterhin danke ich meinen Kolleginnen und Kollegen, die durch ihre Diskussionsbereitschaft zum Gelingen dieser Arbeit beigetragen haben.

Oktober 2002 Yongyu Chang

Kurzfassung

Die innovative Idee, eine von Landessprachen unabhängige, direkte und unverschachtelte Nutzoberfläche auf der Basis intelligenter CAD-Objekte zu entwickeln, macht Bausoftware weltweit nutzbar. Kern dieser Arbeit ist daher das typisch ingenieurwissenschaftliche Ziel, durch entsprechende Entwicklungs- und Forschungsanstrengungen das Werkzeug „CAD-System“ mit integrierten intelligenten Leistungen so auszustatten, dass dadurch für alle späteren Anwender des CAD-Systems die Bedienung einfach und anschaulich wird.

In dieser Arbeit werden daher systematisch Elemente einer Nutzoberfläche gesucht und entwickelt, die kontextsensitiv ein direktes Arbeiten am Objekt erlauben, also ohne verschachtelte Menüfelder und ohne umfangreiche Symbolanhäufungen an den Bildrändern. Mit diesen Elementen werden neue Vorgehensweisen konzipiert, die unter Voraussetzung intelligenten Objektverhaltens effizientestes, eingabearmes Konstruieren ermöglichen. Was dabei „intelligentes Objektverhalten“ ist, wird konkret an ein-, zwei- und dreidimensionalen Objekten diskutiert und an ausgewählten Beispielen programmtechnisch verwirklicht. Auf diese Weise wird die entwickelte Theorie einer von Landessprachen unabhängigen Nutzoberfläche mit intelligenten CAD-Objekten im Experiment ingenieurwissenschaftlich nachgewiesen. Besonders anspruchsvoll ist dabei die Randbedingung, diese Leistung in ein lebendes CAD-System der oberen Leistungsklasse durch Re-Engineering störungsfrei zu integrieren. In weiterführenden Arbeiten können dann, aufbauend auf dem Erfahrungsfundus dieser Arbeit, immer komplexere Aktionen mit bautypischer Intelligenz automatisiert werden, so dass sich die Bedienung vom CAD-System für den Anwender immer einfacher und effizienter gestaltet.

Das Ziel dieser Arbeit ist es, trotz Verzichtes auf sprachgebundene Texte, die unmittelbare Verständlichkeit der hier konzipierten Nutzoberfläche umfassend zu verwirklichen. Aber es liegt in der Natur der Sache, dass in Sonderfällen Unabhängigkeit von Sprache und Schrift unmöglich ist. Die Verwendung von Texten soll jedoch auf unabdingbare Fälle begrenzt bleiben und dabei so gestaltet werden, dass sie möglichst mit Hilfe landessprachlich angepasster Betriebssysteme verständlich werden.

Abstract

The innovative idea of developing an easy-to-use interface based on intelligent CAD objects which does not rely on any specific native language of the user is furthering the global reach and application of unique civil engineering software. The essence of this undertaking is to modify the CAD-system to encompass integrated intelligent functions so as to ensure all potential users a simple and easy handling. Areas of the user interface are developed, enabling a context-sensitive working on the object of desire, eliminating the redundant menus and excess of symbols lining the edges of the screen. Thus via this intelligent object handling new approaches to existing problems can be derived, facilitating an efficient construction without excess interaction. What exactly „intelligent object handling“ incorporates will be discussed and technically shown for chosen examples of one-, two- and three-dimensional objects. This way the developed theory of a language-independent user interface with intelligent CAD-objects is put into a practical context and thus scientifically proven, so to speak. Especially difficult is the boundary condition that this be integrated into an upper class „living“ CAD-system via re-engineering, without any further disturbances. Using this work as a starting point, subsequent strategies may build upon the basics laid here to automatize more and more complex actions, rendering CAD-systems easier to use and more efficient.

The goal of this work is to enable a direct understanding of the user interface developed here despite the forsaking of language-based text. But, of course, there will always be exceptions where an independence of language and text is not possible. The use of text, however, should be restricted to a minimum of cases and be of the kind that they be understandable by adapting the operating system to the native language.

Inhaltsverzeichnis

Bildverzeichnis

Tabellenverzeichnis

1. Einführung

"Aus der Vergangenheit kann jeder lernen.
Heute kommt es darauf an, aus der Zukunft zu lernen. "
[Kybernetiker Hermann Kahn]

Aus der Zukunft des Bauwesens zu lernen heißt zunächst, die zentralen Trends zu erkennen, die sich herauskristallisieren:

- Internationalisierung der Märkte des Bauwesens mit globalem Wettbewerb aller Ressourcen, besonders stark der Bauinformatik,
- Wandel von Bauunternehmen zu Dienstleistungsunternehmen mit Angeboten rund um das Bauen vom Videofilm mit virtueller Realität des geplanten Bauwerks bis zum Facility Management nach dem Bauen,
- Verlagerung der Bautätigkeit von Neubauten in den Bestand,
- Privatisierung öffentlicher Aufgaben.

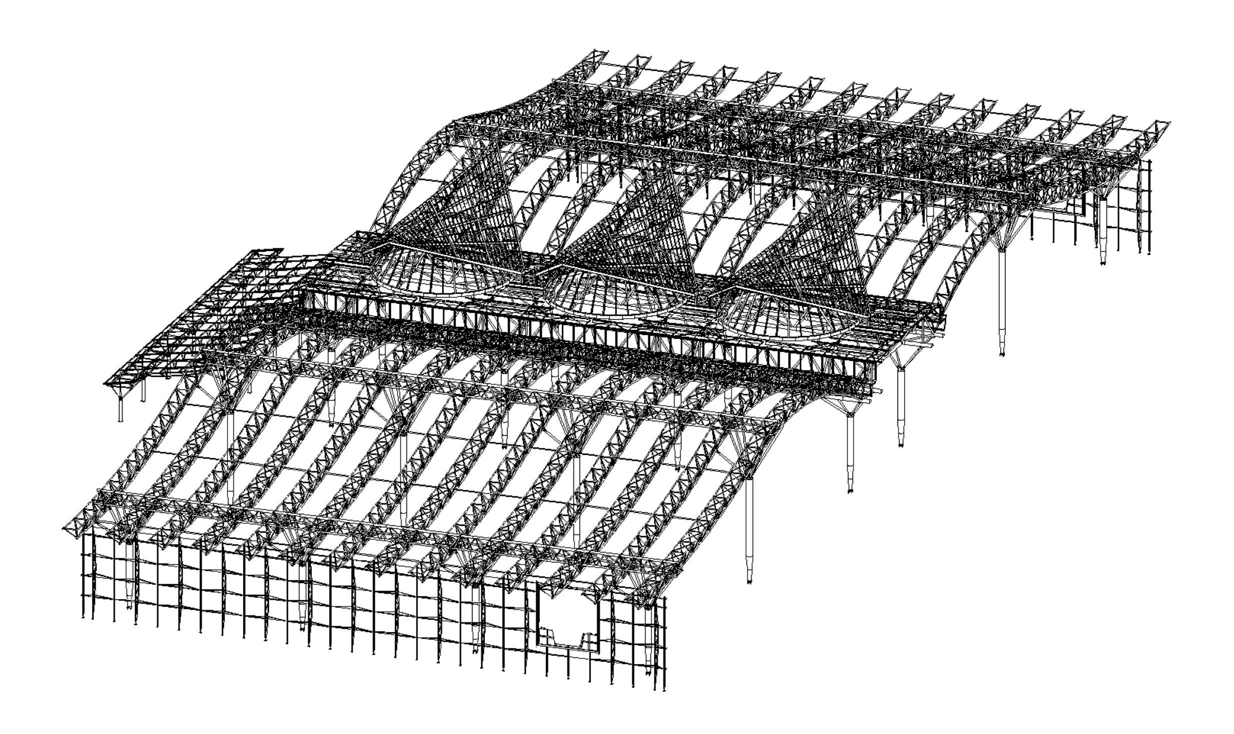


Bild 1-1: CAD-Konstruktion des Zentralbahnhofs Kuala Lumpur

Aus diesen Zukunftstrends [nach 11] sind Visionen und Konzeptionen für das zentrale Softwarewerkzeug der Bauingenieure zu folgern, das rechnergestützte Entwerfen, Konstruieren und Fertigen, kurz CAD genannt. Dieses für vernetztes Arbeiten prädestinierte Werkzeug gewinnt weiter an Bedeutung. Bisher lebten Bauunternehmen und Ingenieurbüros vom produktorientierten Vertriebsansatz. Zukünftig müssen sie umstrukturieren zur dienstleistungsorientierten Organisation als virtuelle Unternehmen mit verteilter, kooperativer Arbeit im Netz. In diesem Sinne die Voraussetzungen für Internationalisierung, also weltweite Praxistauglichkeit eines CAD-Systems zu erforschen und zu schaffen, erfordert Zielbewusstsein und Phantasie. Die CAD-Konstruktion begeisternder Bauten weltweit ist Ansporn und lohnendes Ziel der Forschungsanstrengungen, Bild 1-1.

Selbst in Ländern mit hervorragender Bauingenieurausbildung setzen sich in der Baupraxis nicht die anspruchsvollen, sondern die leicht und effizient zu bedienenden CAD-Systeme durch, die schon mit dem ersten Eindruck Laien und Fachleute überzeugen. Ein wesentliches Forschungsziel der Bauinformatik ist deshalb, die Bedienung von CAD-Systemen durch eingebettetes, bauspezifisch intelligentes Verhalten einfach zu gestalten, selbst bei komplexen Aufgabenstellungen. Entscheidender Schlüssel, ob Forschung zu Computeranwendungen auch zum Erfolg in der Praxis des Bauwesens führt, ist also die Einfachheit der Nutzoberfläche. „Einfache Bedienung durch künstliche Intelligenz“ ist ein typisch ingenieurwissenschaftliches Forschungsziel. Für leistungsstarke Branchensoftware ist „Einfachheit“ zudem für globalen, weltweiten Einsatz eine Voraussetzung. Ideale Eigenschaft dazu wäre die Unabhängigkeit der Bausoftware von Landessprachen. Diese Ziele, Einfachheit und Unabhängigkeit von Landessprachen, scheinen erreichbar. Eine Kombination von Ansätzen, die gemeinsam den Arbeitsaufwand der CAD-Konstrukteure minimal halten, ist dazu erforderlich. Klassische Ansätze allein erreichen dieses Ziel nicht, wie die Bilder 1-2 und 1-3 belegen.

Bei klassischem Ansatz werden ohne Kontextsensitivität unter hohem Platzverbrauch auch Symbole sichtbar angeboten, die im Aktionszusammenhang nicht relevant sind. Bild 1-3 zeigt die unübersichtliche Wirkung von permanenten Symbolanhäufungen. Die Häufigkeit von Mausbewegungen vom dargestellten Objekt zum gesuchten Symbol und zurück bei der baupraktischen CAD-Alltagsarbeit

ist zu erahnen.

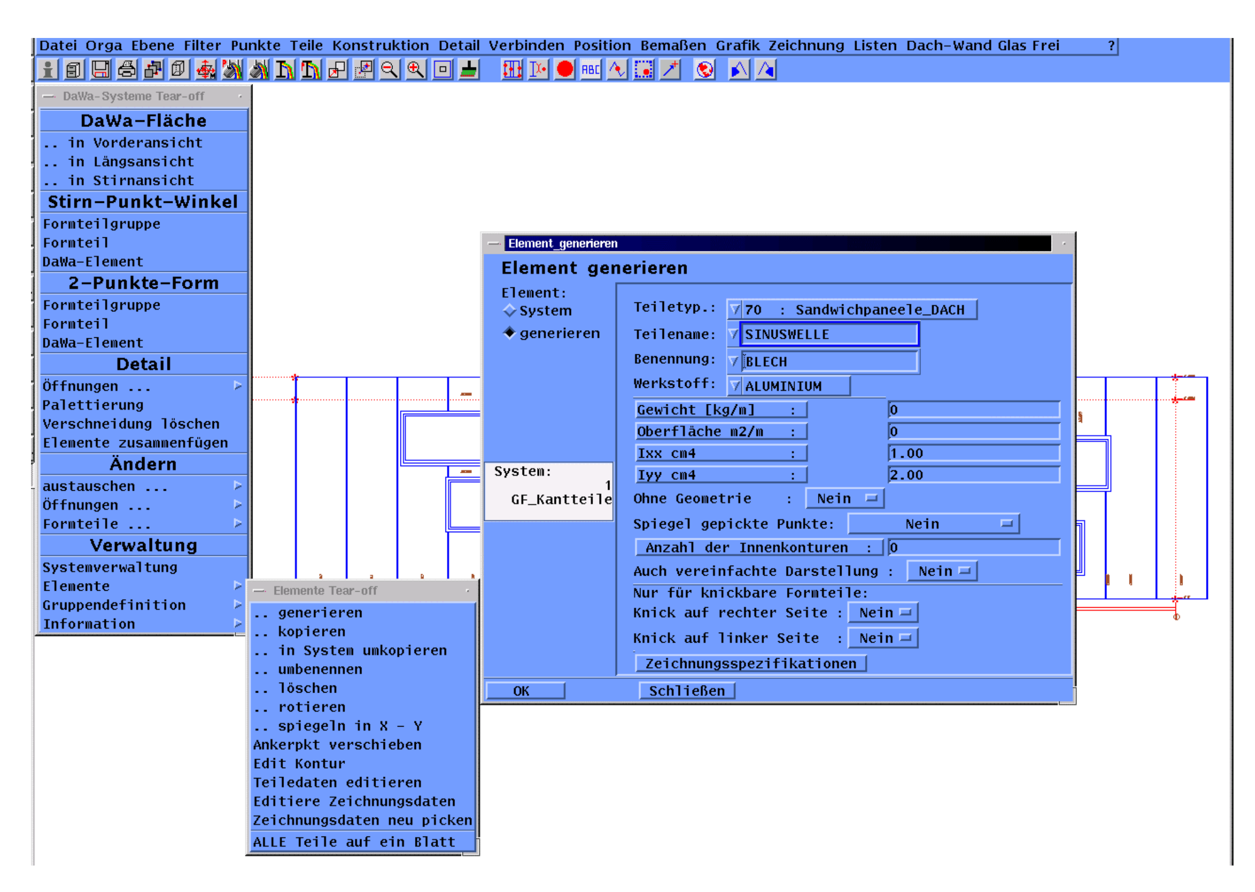


Bild 1-2: Klassischer Ansatz: Unübersichtliche, arbeitsaufwendig tiefe Verschachtelung

In dieser Arbeit werden daher systematisch Elemente einer einfachen Nutzoberfläche gesucht und entwickelt, die kontextsensitiv ein direktes Arbeiten am Objekt erlauben. Klassische Ansätze werden in einem großen Entwicklungsschritt ersetzt durch anschaulichere und effizientere Ansätze auf Basis modernster Softwaretechnologie. Die gravierenden Nachteile klassischer Ansätze mit tief verschachtelten Menüfeldern (Bild 1-2) und umfangreichen permanenten Symbol-anhäufungen an den Bildrändern (Bild 1-3) werden vermieden.

Das grundsätzliche Problem, ohne Dateneingabe Objekte nicht nur qualitativ, sondern quantitativ ohne handwerkliche Präzisionsanforderungen exakt zu platzieren, wird durch Objektfang und Einrasten an wahrscheinlichen Platzierungs-orten gelöst, die sich in der aktuellen Umgebung selbständig anbieten.

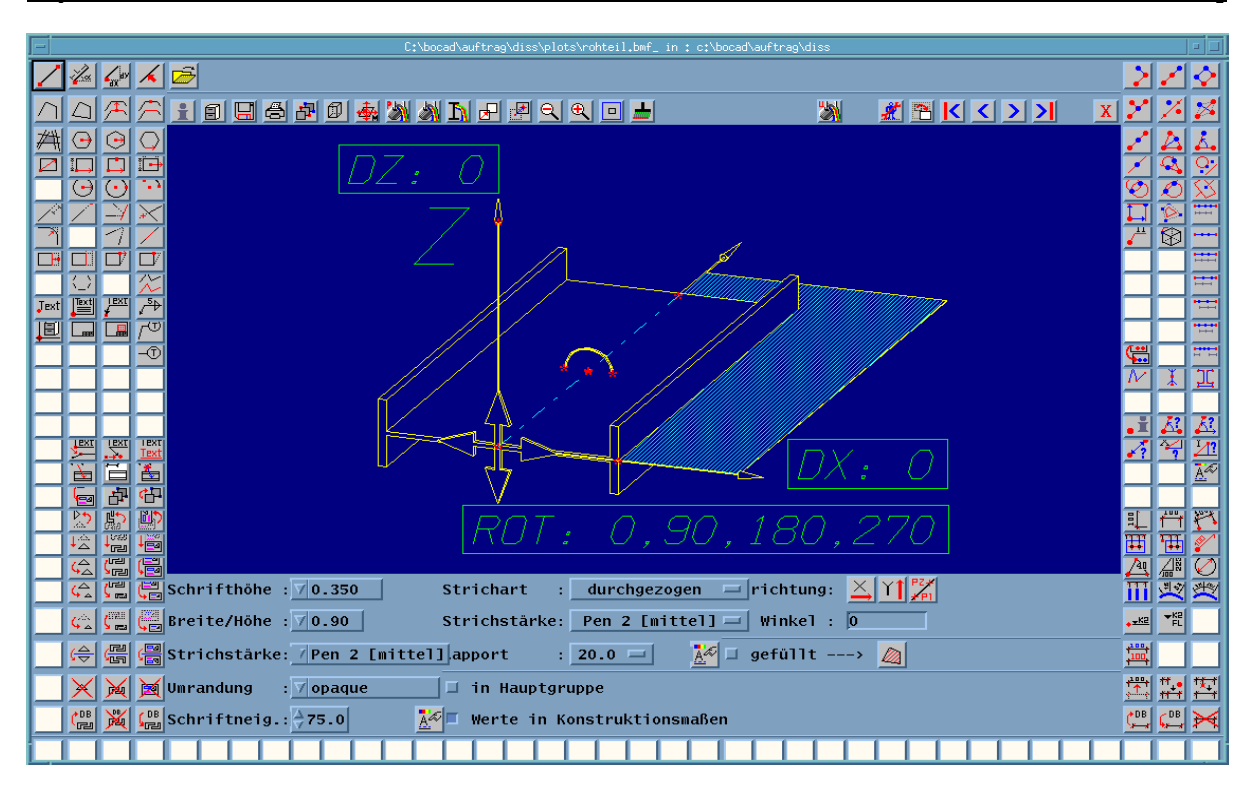


Bild 1-3: Klassischer Ansatz: Überfrachtung durch permanente Symbolanhäufung

Wesentliches Element der konzipierten Nutzoberfläche sind textfreie, selbsterklärende bzw. allgemein bekannte „Funktionsgriffe“ direkt an Objekten, kombiniert mit zugehörigen Methoden für Funktionen mit objektspezifisch intelligentem Verhalten. Funktionsgriffe machen interaktive Aktionen, z. B. Verlängern eines Profilstabes durch Ziehen an seinem stirnseitigen Griff, außerordentlich einfach und direkt. Selbst für komplexe Aktionen hat der CAD-Ingenieur am Bildschirm anstelle vieler einzelner Lösungsschritte lediglich die gewünschte Aufgabe über Wählen und Ziehen des passenden Griffs direkt am intelligenten Objekt zu stellen. Jedes Objekt bietet für die mit ihm möglichen Aktionen, z. B. Verschweißen, Verschrauben, Verschieben, Kopieren, Löschen etc., aufgabentypische Griffe an. Die entsprechende Lösung liefert dann das CAD-System automatisch durch intelligentes, branchentypisches Objektverhalten. Die Idee, auf Ziehen an Funktionsgriffen intelligent reagierende Objekte zu entwickeln, ist kontinuierlich auszubauen von einfachen Objekten des Bauwesens, wie Trägern und Blechen, bis hin zu hochkomplexen Objekten, wie kompletten Treppentürmen. Die vorliegende Arbeit leistet einen Beitrag, diesem hochgesteckten Ziel nahe zu kommen und so die Baupraxis weltweit weiter zu bringen. Europäische

Hochtechnologie auf diese Weise von Landessprachen unabhängig zu machen und im Heimatland China des Verfassers einzusetzen, ist ein besonderes Anliegen der Arbeit.

Mit direkten, kontextsensitiven Elementen werden neue Vorgehensweisen konzipiert, die unter Voraussetzung intelligenten Objektverhaltens effizientestes, eingabearmes Konstruieren ermöglichen. Was dabei „Intelligentes Objektverhalten" ist, wird konkret an ein-, zwei- und dreidimensionalen Objekten diskutiert und an ausgewählten Beispielen programmtechnisch verwirklicht. Auf diese Weise wird die entwickelte Theorie einer von Landessprachen unabhängigen Nutzoberfläche mit intelligenten CAD-Objekten im Experiment mit dem zur Verfügung gestellten System BOCAD-3D ingenieurwissenschaftlich nachgewiesen. Besonders anspruchsvoll ist dabei die Randbedingung, diese Leistung in ein „lebendes" CAD-System der oberen Leistungsklasse durch Re-Engineering störungsarm zu integrieren. In weiterführenden Arbeiten können dann, aufbauend auf dem Erfahrungsfundus dieser Arbeit, immer komplexere Aktionen mit bautypischer Intelligenz automatisiert werden, so dass sich die Bedienung von CAD-Systemen für den Anwender immer einfacher und effizienter gestaltet.

2. Stand der Forschung und Vorarbeiten

Technologietransfer durch Doktoranden des Lehrstuhls Stahlbau der Ruhr–Universität Bochum ermöglichte 1983 die Gründung des Software-Entwicklungs-unternehmens BOCAD. Das entwickelte CAD-System wird im Stahlbau, Metallbau, Glas- und Holzbau bezüglich der Kernleistungen als technisch führendes System der oberen Leistungsklasse angesehen und imitiert. Die thematischen Anregungen zu dieser Arbeit und der Wissenshintergrund für die programmtechnische Realisierung stammen aus der Praxis dieses Unternehmens. Die Erfahrung des Forschungs- und Entwicklungsteams ist eine hervorragende Ausgangsbasis und sichert den Praxisbezug. Weltweit sind etwa 2000 Lizenzen dieses CAD-Systems im Alltagseinsatz kleiner, mittlerer und großer Konstruktionsbüros, die Bauwerke von einfachen Stahltragwerken bis hin zu filigranen Stahl/Glas Konstruktionen höchsten Schwierigkeitsgrades konstruieren. Gebäude von Weltrang werden mit BOCAD-3D konstruiert, Bild 2-1.

Die Nutzoberfläche dieses in den Kernleistungen kontinuierlich weiterentwickelten Systems stößt inzwischen an die typischen Grenzen bisheriger CAD-Systeme. Die zu komplexe Bedienung verhindert, dass im Bauwesen das Nutzungspotenzial des rechnergestützten Konstruierens voll ausgeschöpft werden kann.

Bedingt durch das in Bild 2-1 veranschaulichte, sehr umfangreiche Leistungs-vermögen, gerade für extreme Aufgabenstellungen sowie die weitgefächerte firmen-spezifische Anpassbarkeit, ist die klassische Nutzoberfläche des CAD-Systems umfangreich und verschachtelt. Es hat auf dem Weltmarkt daher den Ruf, das zwar leistungsfähigste und kompletteste System im Stahl-, Metall-, Holz- und Glasbau zu sein, aber auch gleichzeitig schwierig zu bedienen. Der volle Leistungsumfang, die zweckmäßigen Arbeitsstrategien sowie Umfang und Verschachtelungstiefe der Menüs und Symbole der Nutzoberfläche sind nur nach umfangreicher Schulung überschaubar und effizient einzusetzen.

Ein starkes Hemmnis der globalen Verbreitung ist zudem der hohe Aufwand, halb-jährlich jede neue Version mit Leitmeldungen und Dokumentation in vielen Sprachen der Welt zu versehen.

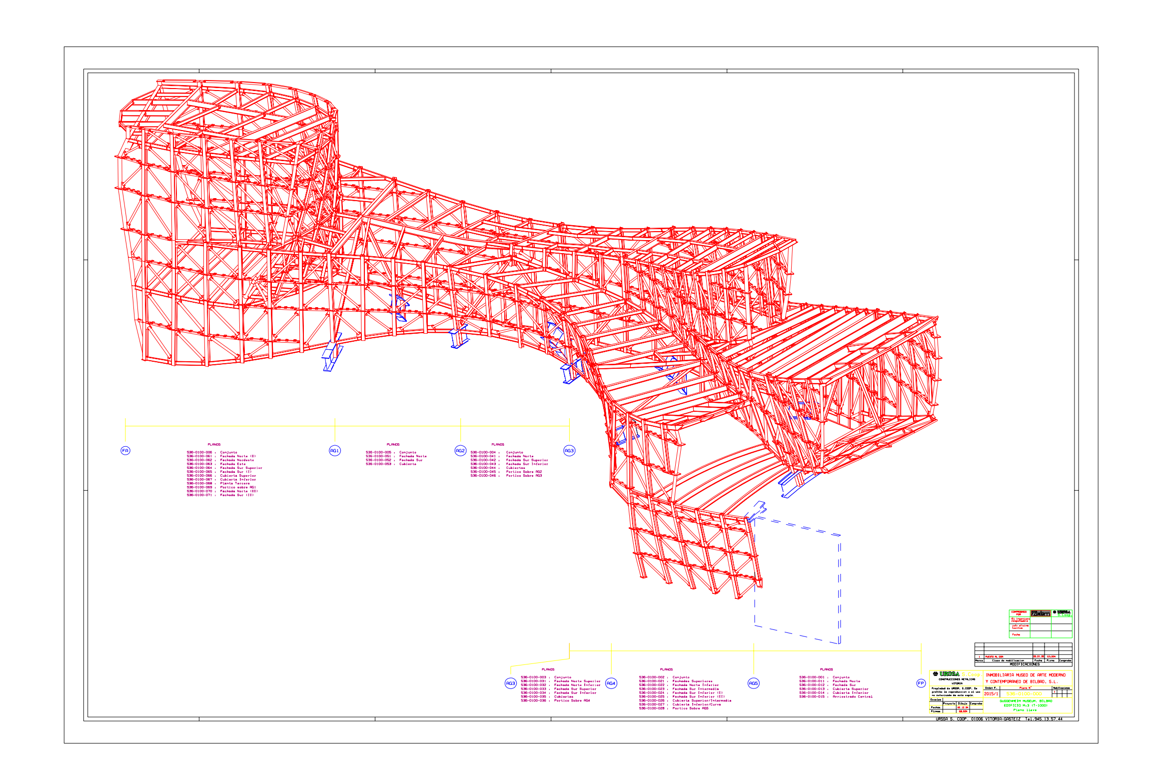


Bild 2-1: CAD-Modell des Stahltragwerks „Guggenheim Museum Bilbao“

Bei CAD-Systemen mit hohen Anforderungen an den Anwender bleibt also der Erfolg begrenzt auf die Bauunternehmen und Konstruktionsbüros, die hauptsächlich architektonisch anspruchsvolle Bauwerke erstellen und ein entsprechend leistungsfähiges, überlegenes System auch noch dann akzeptieren, wenn die Nutzoberfläche höhere Anforderungen an die CAD-Konstrukteure stellt. Dem allgemeinen Trend zu komplexer, filigraner Stahl- und Glasarchitektur müssen aber mehr und mehr alle Bauunternehmen und Konstruktionsbüros folgen. Der Massenmarkt der Stahl-, Holz- und Glasbaubranche in Deutschland und weltweit fordert also, die Nutzoberfläche von CAD-Systemen durch entsprechende Forschung deutlich einfacher, selbsterklärender und direkter zu gestalten. Im Vergleich zu anderen Technologien, z. B. auf Autocad Basis mit auf Autocad - Elemente begrenzter Nutzoberfläche, kann in dieser Arbeit durch Eigenentwicklung der Gesamtsoftware die Nutzoberfläche völlig frei und damit beliebig einfach gestaltet werden. Für die Forschung ist es somit ein wesentlicher Vorteil, dass das zur Verfügung gestellte CAD-System vollständig ohne unterlegte Basissoftware

entwickelt wurde, so dass hier die Nutzoberfläche tatsächlich innovativ gestaltet werden kann.

Die Anwender sind interessiert, als neue Entwicklung die hier konzipierte, um eine Größenordnung effizientere und gleichzeitig leicht zu bedienende Nutzoberfläche für ihre Alltagsarbeit zu erhalten. Diese Unternehmen können dann ihrerseits durch effizienteres und profitableres Arbeiten ihren Marktanteil ausbauen und entsprechend die Zahl ihrer qualifizierten Bauingenieure erhöhen. Allein innerhalb des deutschsprachigen Raums werden ca. 1500 Anwender die Forschungsergebnisse zur Steigerung ihrer Leistungsfähigkeit nutzen. Das bestehende Netzwerk der Anwender wird kontinuierlich weiter ausgebaut, um den Erfolg der Entwicklungen sicherzustellen. Nach Gründung eines Büros zur Unterstützung in Polen wurde gemeinsam mit dem modernsten Stahlbauunternehmen Ungarns auch ein Büro in Budapest gegründet. Weiterhin besteht ein Büro in Kuala Lumpur für Ostasien und speziell für Glasbau in England. In diesem Sinne kann die vorliegende Arbeit auch dazu dienen, Wissenstransfer und CAD-Hochtechnologie in China zu verbreiten.

3. Ingenieurwissenschaftliche Ziele, Randbedingungen, Einschränkungen

3.1 Nutzoberfläche und branchenspezifische Intelligenz

Die innovative Idee, eine von Landessprachen unabhängige, direkte und unverschachtelte Nutzoberfläche auf der Basis intelligenter CAD-Objekte zu entwickeln, macht Bausoftware weltweit nutzbar. Die Projektidee vereinfacht die Bedienung so, dass neben der bisherigen technologischen Führerschaft in den Kernleistungen sich eine weitere weltweite Verbreitung im Bauingenieurwesen auch dort erschließt, wo bezüglich der Ausbildung und der Wissensvoraussetzungen der Anwender keine weit überdurchschnittlichen Anforderungen gestellt werden dürfen. Kern dieser Arbeit ist daher das typisch ingenieurwissenschaftliche Ziel, durch entsprechende Entwicklungs- und Forschungsanstrengungen das Werkzeug „CAD-System“ mit integrierten intelligenten Leistungen so auszustatten, dass dadurch für alle späteren Anwender des CAD-Systems die Bedienung einfach und anschaulich wird.

Heutige CAD-Systeme für das Bauwesen sind über viele Jahre gewachsen und an der Alltagspraxis gereift. Der Forschungs- und Entwicklungsaufwand einzelner CAD-Systeme umfasste im Laufe der Zeit zumeist einige hundert Mannjahre. Die Anwender, oft viele Tausende, haben insgesamt in Milliardenhöhe in Ausbildung, Fortbildung, Wartung, Lizenzgebühren und Hardware investiert. Diese Systeme mit einem revolutionären Ansatz völlig neu zu entwickeln, wurde gegen die ingenieurwissenschaftlichen Ziele der wirtschaftlichen Machbarkeit und Marktakzeptanz verstoßen. Daher werden im Rahmen dieser Arbeit keine revolutionären, sondern evolutionäre Ansätze entwickelt, die in verkraftbaren Schritten zu einem sequentiellen Re-Engineering von CAD-Systemen führen.

In diesem Sinne gilt auch das Thema „Eine von Landessprachen unabhängige Nutzoberfläche mit Intelligenten CAD Objekten des Bauwesens“ unter der klaren Einschränkung, nur ingenieurwissenschaftlich sinnvolle Teile der CAD-Systeme von Texten und damit von Landessprachen und -schriften unabhängig zu machen. Die unterschiedlichen Betriebssysteme Windows und LINUX, in die die CAD-Systeme als Anwendungsprogramme eingebettet sind, sind in aller Regel sowohl an die

Landessprache als auch an die landesüblichen Schriftzeichen angepasst. Die Windows-typischen hierarchischen Listen der Baumstruktur des Dateisystems, Listenfelder mit den bekannten Profilnamen und Werkstoffen, aber auch Kommentare, die der CAD-Ingenieur Objekten anfügt, bleiben selbstverständlich textförmig. Hierbei entsteht kein Zusatzaufwand und insbesondere auch kein Wiederholungsaufwand bei jeder weiteren Programmversion. Gleiches gilt für Kurzinformationen (Quickinfo), die frei von Grammatik durch landesspezifische Übersetzungstabellen mit Einmalaufwand eingebunden werden. Lediglich für neue Leistungen mit entsprechenden Kurzinformationen sind die bestehenden Übersetzungstabellen zu erweitern.

Letztlich werden textbasierte Lösungen auch für alle restlichen Fälle belassen, für die keine wirklich selbsterklärende, sprachfreie grafische Lösung gefunden werden konnte. Die Onlinehilfe beispielsweise mit Erläuterungen und Schulungsbeispielen kann sicher nie vollständig auf Symbole und Bilder beschränkt werden.

Es bleibt bei CAD-Hochleistungssystemen jedoch noch reichlich Spielraum, um sinnvolle, besonders häufig genutzte Leistungsanteile sprachfrei grafisch zu gestalten, wie die folgenden Abschnitte zeigen.

Die Nutzoberfläche von branchenspezifischen CAD-Systemen kann nur dann einfach gehalten werden, wenn fachlich intelligentes Verhalten der konstruierten Objekte dies gestattet. Die Einfachheit der Bedienung wird also erst ermöglicht durch entsprechend in das CAD-Programm eingebettete Intelligenz. Beides bedingt einander. Konzeptionelle Gestaltungsideen zur Oberfläche sowie entsprechende CAD-Methoden für intelligentes Verhalten bautypischer CAD-Objekte und CAD-Funktionen sind daher die wesentlichen Inhalte dieser Arbeit. Der Ansatz für direkte Nutzoberflächen wird mit innovativen Gestaltungsideen spezifisch auf CAD-Systeme des Stahl-, Holz- und Glasbaus angewendet, deren Objekte mit bauspezifischer Logik intelligent auf Aktionen des CAD-Konstrukteurs reagieren. Erst in der Kombination einer direkten Nutzoberfläche mit intelligent reagierenden CAD-Objekten entsteht eine Weltneuheit für den Stahl-, Metall-, Holz- und Glasbau in voller Tragweite mit marktentscheidendem Nutzen. Es gelingt dadurch, trotz hoher Komplexität in der Sache die Bedienung einfach, verständlich und direkt, also

effizient zu gestalten. Intelligente Reaktionen von CAD-Objekten sind bis zu hochkomplexen Konstruktionsproblemen denkbar. Die Änderung des Höhenniveaus eines Geschosses im Bauwerk hat beispielsweise Folgen bis in die Details der gesamten Treppengeometrie der an das Geschoss anschließenden Treppen. Ob auch dieser höchste Schwierigkeitsgrad im Rahmen dieser Arbeit gelöst werden kann, bleibt zu untersuchen.

Das Projekt erfordert Gestaltungsideen, welche selbsterklärenden Griffe an welchen Stellen die verschiedenen Objekte des Stahl-, Holz- und Glasbaus für welche Aktionen jeweils anbieten sollen. Mit welchen bauspezifisch intelligenten Reaktionen diese Objekte auf Griffaktionen des Benutzers antworten, ist dazu passend zu entwickeln. Zumindest alle häufig in der Praxis genutzten Funktionen und ihre Nutzoberfläche werden effizient und anschaulich zu gestalten sein. Die programmtechnische Umsetzung einer derartigen Nutzoberfläche mit intelligenten Objektreaktionen sowie die Gestaltungsideen zu Objekten mit Griffen müssen von Anfang an die aus Sicht der späteren Anwender praxisgeeigneten Lösungswege berücksichtigen.

Eine direkte Nutzoberfläche kommt ohne umständliche, tiefe Menüverschachtelung und Flut von Symbolen aus. Sie ist also schon dadurch entscheidend effizienter als klassische Lösungen. Durch selbsterklärende „Griffe" (Bild 3-1), die unmittelbar an intelligent reagierenden CAD-Objekten angebracht sind, wird die Bedienung nahezu banal. Vorab soll ein besonders einfaches Beispiel aus dem Stahlbau diesen Sachverhalt belegen, Bild 3-1.

Das Verlängern bzw. Verkürzen des Objekts „Profilstab" an einer seiner Stirnseiten geschieht durch Ziehen des entsprechenden Griffs direkt an der gewählten Stirnseite, bis das gewünschte Verlängerungsmaß erreicht ist. Das Maß wird als Statuszeile laufend angezeigt. Auch der Träger reagiert anschaulich mitlaufend.

Die häufige Aktion „Verlängern/Verkürzen" ist somit auf die effizienteste, kürzestmögliche Aktion reduziert, insbesondere in Kombination mit der Funktion „Einrasten".

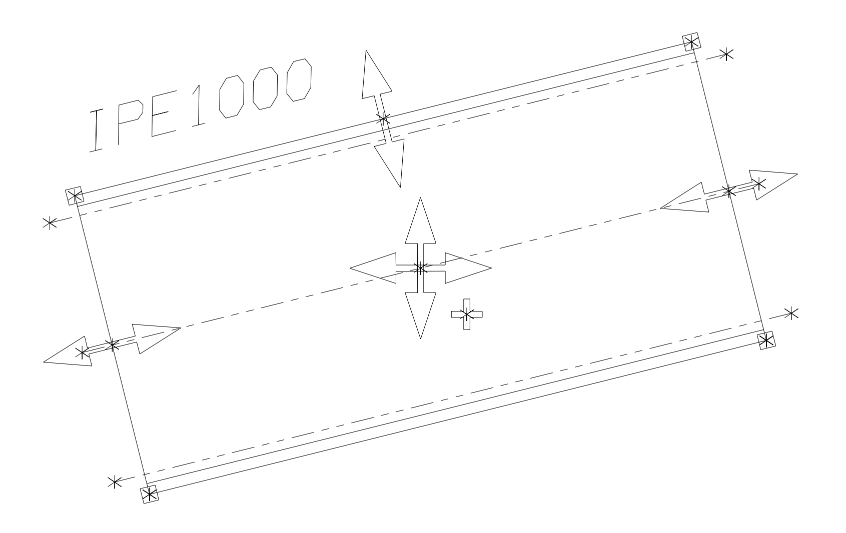


Bild 3-1: Griffe für Stabverlängern, Profiländern, Verschieben, Kopieren

Ein zusätzlicher Dehn- und Stauchgriff auf der Stabachse gestattet ein Stauchen oder Dehnen des Trägers unter Beibehaltung aller seiner Details. Gestaltet man die Verlängerungsgriffe verschiebbar von der Stirnseite an einen anderen Ort auf der Stabachse, an dem Dehnen oder Stauchen erwünscht ist, steigt der Leistungsumfang sogar ohne zusätzliche Symbole.

Zieht man den Profilgriff quer zur Längskante des Trägers, so wird das bisherige Profil gegen das jeweils nächste Profil der gleichen Profilreihe ersetzt. Dabei werden die ursprünglichen Lageparameter des Trägers automatisch beachtet, d. h. ein bündig verlegter Träger bleibt bündig. Auch die logischen, konstruktiven Konsequenzen der Aktion werden beachtet: Es bietet sich für eine Nebenträger an, der an einen Hauptträger über Doppelwinkel oder Kopfplatten angeschlossen ist, den Anschluss und die Schraubenanordnung automatisch der so geänderten Profilgröße anzupassen. Es werden also nicht nur die bautechnischen Schlussfolgerungen gezogen, dass ein Profil nur in diskreten, lieferbaren Profilsprüngen geändert werden kann, sondern auch die wesentlich komplexeren Schlussfolgerungen, welche Konsequenzen die Änderung auf die umgebenden Teile hat. Dies erfordert erhebliche Bauingenieurintelligenz der CAD-Objekte, die im Projekt erdacht und entwickelt werden soll. Da Profiländerungen mit allen damit verbundenen Konsequenzen in den Konstruktionsbüros häufig vorkommen, ist die effizienteste Lösung dieses Problems von erheblichem wirtschaftlichen Nutzen.

Das Objekt „Blech“ bietet in blechtypischer Weise ähnliche Griffe an, mit denen die Blechkontur unabhängig von rechnerinternen Problemen (Lage des lokalen Koordinatensystems etc.) über direktes Ziehen des gewählten Griffs beliebig geändert werden kann. Griffe zum Ändern der Blechdicke, Verschieben und Kopieren von Blechen haben die gleiche Form und Funktion wie bei Profilstäben, um den Wiedererkennungswert zu wahren und so zusätzlichen Lernaufwand zu vermeiden.

3.2 Re-engineering großer Bausoftwaresysteme mit objektorientierter, datenbankbasierter Technologie

Große und sehr große Bausoftware-Systeme, die über viele Jahre im Praxiseinsatz ausgereift sind und sich mit großem Markterfolg bewährt haben, bleiben nicht automatisch nachhaltig technisch führend. Sie sind grundsätzlich wie alle Ingenieurprodukte einem Lebenszyklus unterworfen, der mit dem Ersatz durch spätere, aber technisch und wirtschaftlich bessere Konkurrenzprodukte enden kann. Im Gegensatz zu materiellen Produkten, die trotz Wartung bis zur wirtschaftlichen oder technischen Unbrauchbarkeit altern, kann aber gerade Software als immaterielles Produkt laufend durch Wartung und Weiterentwicklung lange auf höchstem technischen Stand gehalten werden. Dennoch sind besonders in der Informatik über Jahrzehnte ausgeprägte Technologiesprünge zu beobachten, die im Prinzip, im Ausmaß oder in den wirtschaftlichen Implikationen unvorhersehbar waren und folglich in der ursprünglichen Konzeption großer Bausoftware-Systeme nicht berücksichtigt sind. Auch das Umfeld der Anwendung kann ebenso große Sprünge aufweisen, z. B. durch paralleles, verteiltes Arbeiten im Netz oder vollständige Integration von kaufmännischer und technischer Software eines Unternehmens. Ohne geeignete Gegenmaßnahmen, u. a. auch zur Reduzierung des Entwicklungs- und Wartungsaufwands, führt dies dann doch zum wirtschaftlichen Ende großer, bewährter Softwaresysteme und ggf. auch des dahinter stehenden Softwareunternehmens.

Es ist daher eine bedeutende, typisch ingenieurwissenschaftliche Aufgabenstellung, übertragbare und reproduzierbare Lösungen zur Erneuerung großer, ausgereifter Software-Systeme zu erforschen und im Versuchsfeld anhand typischer Fälle

nachzuweisen.

Bei kleineren Systemen ist ein vollständiges Re-design mit jeweils neuesten Werkzeugen der Informatik parallel zum Alltagseinsatz des zu ersetzenden Systems wirtschaftlich und technisch eine sinnvolle Lösung. Bei sehr großen Systemen mit entsprechendem Leistungsumfang, in die mehrere hundert Mannjahre Entwicklung zu investieren waren und die große Ingenieurteams zur laufenden Wartung erfordern, ist dieser Weg ein extremes unternehmerisches Wagnis, das über viele Jahre ein existentielles Risiko darstellt.

In dieser Arbeit soll daher untersucht werden, ob und wie große Bausoftware-Systeme bei laufendem Alltagseinsatz sanft, d. h. für den Nutzer wirtschaftlich und technisch nicht negativ merklich, durch Re-Engineering auch auf unvorhergesehene, große Technologiesprünge der Informatik und des Umfelds der Anwendung umgestellt werden können.

An Technologiesprüngen, deren Konsequenzen in dieser Arbeit zu untersuchen sind, seien genannt:

- Gestaltung der Nutzoberfläche, die durch neue Grafikstandards und hohe Grafikleistung ingenieurtypischer Rechner denkbar wird.
- Wirtschaftliche Verfügbarkeit und technische Möglichkeiten neuester, objektorientierter Datenbanktechniken.
- Entwicklungswerkzeuge, insbesondere plattformübergreifende Werkzeuge.

An Sprüngen im Umfeld der Anwendung seien genannt:

- Integration neuer Anwendungsfelder, die bisher mit verschiedenen Spezialsystemen bearbeitet wurden, zu einem einzigen, durchgängigen Komplettsystem.
- Bearbeitung eines Konstruktionsauftrags an mehreren Standorten gleichzeitig, z. B. mit externen Konstruktionsbüros im Netz, um kürzeste Termine einzuhalten.
- Integration von Konstruktionssystemen direkt, ohne menschlich gesteuerten

Datentransfer, in die Projektplanung und -steuerung einschließlich Materialwirtschaft, Ablaufplanung und Dokumentenmanagement.

Auch gekoppelte Sprünge sind zu untersuchen, z. B. Technologiesprung der Nutzoberfläche und gleichzeitig Sprung des Einsatzspektrums durch Integration der architekturtypischen Entwurfs- und Konstruktionsprozesse in ein ausgereiftes CAD-System für die Detaillierung von Stahltragwerken und entsprechender Hüll-konstruktion.

Der besondere ingenieurwissenschaftliche Reiz dieser Arbeit liegt auch darin, die sprachfreie Nutzoberfläche mit intelligenten Objekten so zu konzipieren, dass diese Leistungen in ein CAD-System bei laufendem Betrieb mit Re-Engineering Maß-nahmen integriert werden können. Die dabei gemachten Erfahrungen und Schritte zur Problemlösung werden in den Kapiteln der einzelnen intelligenten Objekte beispielhaft erläutert.

Sinn der objektorientierten Programmierung moderner Softwaresysteme erläutern anhand [17]. Re-Engineering des untersuchten CAD System ermöglicht die Transformation von gewachsenen Applikationen in eine objektorientierte und somit leichter wartbare und für die Wiederverwendung besser geeignete Architektur. Folgende neue Objekte werden definiert:

Katalog (Sammlung projektunabhängiger Daten):

Ein Katalog fasst zusammengehörige Daten aus einem Themenbereich zusammen. Er kann hierarchisch aus Unterkatalogen aufgebaut sein. Letztlich enthält ein Unterkatalog eine oder mehrere Katalogobjektgruppen. Der Katalog trägt Kennzeichen der verwendeten Maßeinheiten für Längen, Kräfte, Massen etc. Der Katalog ist insgesamt versioniert, d.h. trägt eine globale Versions „nummer“ für seinen gesamten Inhalt. Er beinhaltet jeweils noch alle vorherigen Versionsstände. Der Katalog ist lizensiert, u. U. für jede Version getrennt.

Katalogobjekt (aus dem Katalog auswählbare Einheit):

Dem Katalogobjekt sind Eigenschaften, die Katalogobjektattribute mit ihren Werten, zugeordnet. Diese Eigenschaften können abgefragt werden. Katalogobjektattribute

bilden am Katalogobjekt eine Katalogobjektstruktur. Diese erlaubt je nach Interpretation z. B., dass ein Katalogobjekt auf ein anderes verweisen kann oder dass ein Katalogobjekt andere beinhaltet, d. h. aus anderen Katalogobjekten aufgebaut ist. Katalogobjekte mit identischer Katalogobjektstruktur sind üblicherweise in Katalogobjektgruppen zusammengefasst.

Beispiele :

Profilquerschnitte

Schraubgarnituren

Zukaufteile

Baugruppen

Katalogobjekte sind nicht individuell, sondern über den Katalog versioniert. Das heißt, dass das Objekt in verschiedenen Katalogversionen identisch sein kann, jedoch zwei verschiedene Versionen eines Objekts auf jeden Fall nur in verschiedenen Katalogversionen enthalten sein können. Katalogobjekte sind innerhalb des Katalogs eindeutig über ihren Namen zu finden. Das heißt, es darf keine gleichnamigen Objekte in unterschiedlichen Katalogobjektgruppen geben.

Designobjekt (Ein Datenobjekt im Projekt)

Auch wenn die exakte Definition für die Anwendung nicht interessiert, ist der Unterschied zum Katalogobjekt wichtig: Das Katalogobjekt stellt lediglich die allgemeine Beschreibung oder auch einen Typ für ein tatsächlich im Projekt verwendetes Objekt bereit. Nur das Designobjekt repräsentiert tatsächlich einen konkreten Gegenstand oder Sachverhalt im Projekt.

Bild 3-2 zeigt das konzipierte API[1].

----

[1] API: **A**pplication **P**rogramming **I**nterface. Schnittstelle eines Betriebssystems oder einer Benutzeroberfläche. Das API stellt Routinen und Dienste bereit, die von anderen Programmen genutzt werden können. So wird die Anpassung an die Hardware erleichtert.

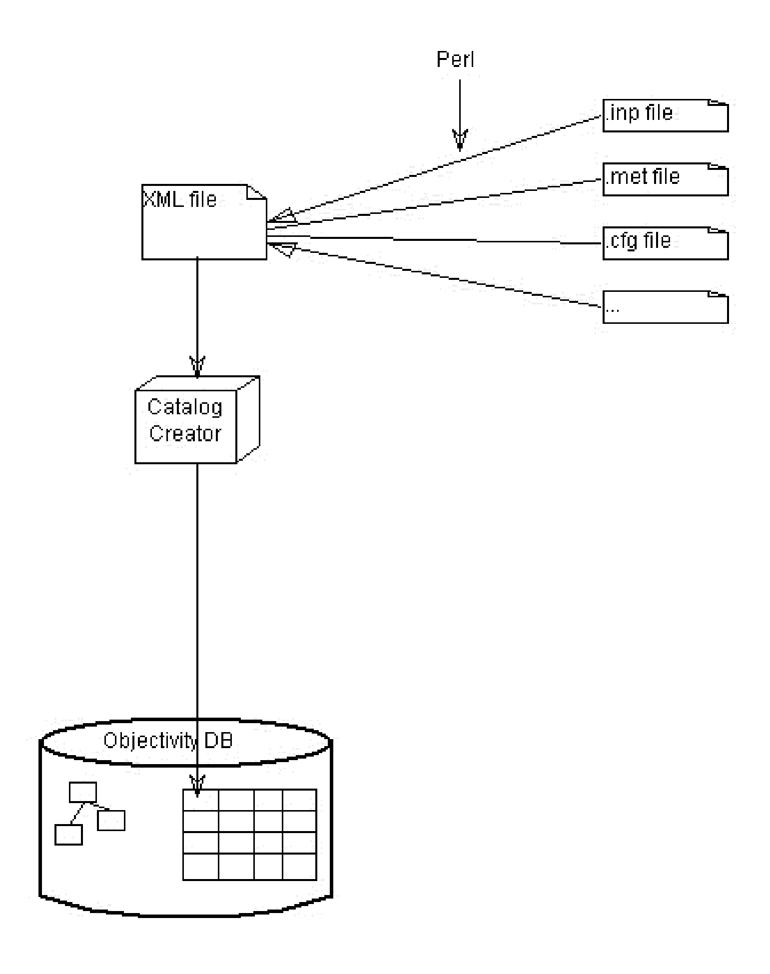


Bild 3-2: Das konzipierte API für Kataloge

Datenbanksysteme kennen die Struktur der Daten, die mit ihrer Hilfe verwaltet werden. Diese Struktur wird häufig mit den Begriffen **Schema** oder auch **Metadaten** bezeichnet, wobei Schema die lesbare Entwurfsform und Metadaten die datenbank-interne Form bezeichnen.

Die Anwendung, die mit dieser Datenbank zusammenarbeitet, benutzt ein **Datenmodell**. Bei objektorientierten Systemen entsprechen sich Datenmodell und Schema im Wesentlichen. Innerhalb von C++ arbeiten das CAD- und das Datenbanksystem mit den »gleichen« Klassendefinitionen (Datenbankschema == C++ Klassenmodell).

Während des Entwicklungsprozesses entstehen die Metadaten aus den Schemata. Diese sind wie normale Datenbestände zu betrachten und auch zu pflegen. Da diese

Metadaten dem DBMS[1] gehören, sind sie zunächst vom konkreten CAD-System losgelöst. Das heißt, dass ein Weg gefunden werden muss, diese Metadaten auch beim späteren Anwender zu installieren und zu pflegen.

Die Metadaten im Entwicklungsprozess zu pflegen heißt, dass diese wie normale Quellen der Versionierung unterworfen sind. Weiterentwicklung des Datenmodells darf zunächst nur aus Hinzufügen neuer Klassendefinitionen bestehen. Datenbanken bleiben in diesem Fall auf- und abwärtskompatibel. Müssen Klassendefinitionen geändert werden, so sollten nur Attribute hinzugefügt werden. In diesem Fall sind Datenbanken durch automatische Evolution aufwärtskompatibel. Auch weitergehende Änderungen sind möglich, allerdings sind dann programmtechnische Migrationshilfen vorzusehen. Besonders in solchen Fällen muss ausführlich dokumentiert werden. Sicherheitshalber sind daher die Schemata und ihre Hierarchie so zu dokumentieren, dass die Metadaten ggf. rekonstruiert werden können.

----

[1] DBMS: Datenbank management System

4. Theoretische Lösungsschritte, Prinzipien und Werkzeuge

4.1 Analyse der Nutzoberfläche von Standardsoftware

In diesem Abschnitt werden die groben Charakteristika der betrachteten CAD-Systeme vorgestellt und die Gründe für ihre Wahl als bekannte de facto Standards oder vorbildliche „Trendsetter“ mit Zukunftswirkung auf Nutzoberflächen erläutert. Die detaillierte Analyse ihrer Nutzoberfläche folgt in späteren Abschnitten, in denen die in dieser Arbeit entwickelte Konzeption für Objekte, ihre Nutzoberfläche und Methoden beschrieben wird. Dadurch sind Wiederholungen zu vermeiden, gleichzeitig wird die Verständlichkeit im Sinnzusammenhang erleichtert.

Im Rahmen dieser Arbeit von besonderem Belang sind die Systeme AutoCAD 2000, Prostahl-3D, Hypersteel, BOCAD-3D, X-Steel, Corel Draw 10, Visio 2000, mb-Programme und ArchiCAD. Zweckmäßige Anregungen, anwenderfreundliche Gemeinsamkeiten und industrieübliche Standards dieser Systeme werden nach Möglichkeit berücksichtigt und mit eigenen Überlegungen weiter entwickelt und systematisiert.

AutoCAD 2000 von Firma AutoDesk ist ein historisch gewachsenes CAD-Basissystem mit stärkster Verbreitung in Europa. Trotz teilweise antiquierter Nutzoberfläche, z. B. MS-DOS Fenster mit Eingabeaufforderung als eine der alternativen Möglichkeiten, ist AutoCAD wegen der Verbreitung ein Marktstandard. Deshalb nutzen viele anwendungsorientierte CAD-Systeme wie ProStahl-3D und Hypersteel im Stahlbau AutoCAD als Basissystem, um darauf aufbauend CAD-Systeme mit höherer anwendungsspezifischer Automatisierung zu entwickeln. Die Bekanntheit von AutoCAD, die in allen Ländern dazu kommerziell angebotenen Schulungen und der Glaube, dass alle AutoCAD Ergebnisse zwischen unterschiedlichen Anwendungen problemlos und ohne Informationsverlust ausgetauscht werden können, sind entscheidende Marktargumente. Für Entwickler von anwendungsspezifischen CAD-Systemen, die auf AutoCAD aufbauen, liefert AutoDesk leistungsfähige Entwicklungswerkzeuge, z. B. Object ARX. Selbst in aktuellen Forschungsarbeiten der Bauinformatik wird deshalb AutoCAD häufig als

Basissystem verwendet [13]. Nachdenklich stimmt, dass AutoCAD sich inzwischen auf das Windows Betriebssystem von Microsoft beschränkt und nur noch den englischen Befehlssatz unterstützt. Entwicklungen wie LINUX werden nicht aufgegriffen. Mit Mechanical Desktop und Architectural Desktop übernimmt AutoDesk nach und nach selbst wesentliche Anwendungsgebiete. Für die zuvor auf diesem Gebiet tätigen Entwicklungsunternehmen kann dies sehr schädlich sein. Die verschiedenen Versionen von AutoCAD machten in der Vergangenheit ohne Rücksicht auf dritte Entwickler inkompatible Entwicklungssprünge mit der Folge, dass auf AutoCAD aufbauende Anwendungssoftware praktisch neu programmiert werden musste. Die existenzbedrohenden Folgekosten derartigen Verhaltens taten erstaunlicherweise der Verbreitung von AutoCAD keinen Abbruch. Es macht aber vor diesem Hintergrund Sinn, nicht von AutoCAD abhängig zu sein, wie z. B. die CAD-Systeme X-Steel und BOCAD-3D, die in der Stahlbaubranche weltweit verbreitet sind. BOCAD-3D ist zudem Betriebssystem-übergreifend konzipiert und daher unter Windows wie unter LINUX nutzbar.

Dem Ziel einer direkten, unmittelbar verständlichen Nutzoberfläche mit modernster Softwaretechnologie kommt das Zeichensystem Corel Draw, Version 8, besonders nahe. Es befruchtete daher ähnlich wie das Betriebssystem des Apple Macintosh die Weiterentwicklungen auch anderer CAD-Systeme und hat dadurch als Trendsetter ähnlich wie VISIO 2000 standardbildende Bedeutung. Beide Zeichensysteme werden daher bei der Auswahl möglichst als bekannt vorauszusetzender Elemente als Referenz stark gewichtet.

Ähnliches gilt für die anwendungsorientierten CAD-Systeme von mb-Programm und ArchiCAD, die besonders für intuitive, mathematisch unbelastete Anwender konzipiert sind. Diesem Anwenderkreis können unanschauliche, komplexe Nutzoberflächen nicht zugemutet werden, die Bauingenieure bisher fast kritiklos als naturgegeben hingenommen haben. ArchiCAD wurde daher hinsichtlich möglicher Vereinfachungen ebenfalls in die Analyse einbezogen.

4.2 Einheitliche Prinzipien einer stufenlos direkten, sprachfreien Nutzoberfläche mit bekannten oder selbsterklärenden Basiselementen

Objektorientierte Programmierung erleichtert durch Objektklassen und Vererbung, bei gut durchdachter Strukturierung einheitliche Prinzipien für die Gestaltung der Nutzoberfläche und der auf die Objekte anwendbaren Methoden zu verwirklichen. Auf alle Objekte von Objektklassen sind mit wenigen Ausnahmen folgende klassenspezifisch geprägten Methoden anwendbar:

Erzeugen, Identifizieren, Informieren, Einrasten, Löschen, Eigenschaften (Attribute) Ändern, Verschieben, Drehen, Kopieren, Klonen, Spiegeln, Detaillieren, Editieren, Erweitern, Reduzieren, Verbinden (Gruppieren), Darstellen, Bezeichnen, Undo, Redo.

Objektorientierte Programmierung allein hilft nicht, die Ideen zu finden, mit denen eine Nutzoberfläche „einfach und unkompliziert“ gemacht werden kann. Sie hilft auch nicht herauszufinden, welche passende Reaktion des Objekts „intelligent“ ist, wenn ein CAD-Anwender über die Nutzoberfläche eine Aufgabe stellt. Dies auszudenken ist der schöpferische Teil dieser Arbeit. Das Ziel, weltweite Einsetzbarkeit ohne sprachliche Anpassung sowie leichte Verständlichkeit durch grafische statt textliche Darstellung zu erreichen, gibt den Rahmen der schöpferischen Ideen vor. Folgende gestalterische Prinzipien folgen daraus und sind einzuhalten:

- Allgemein bekannte oder selbsterklärende Elemente ohne Schulungsbedarf verwenden.
- Passive, aktive und/oder dynamische Grafik sprachübergreifend verständlich gestalten.
- Direkte Aktionen am Objekt vorsehen statt umständlicher Umwege über tiefe Menüverschachtelungen oder Symbolanhäufungen am Fensterrand.
- Analoge Aktionen und Nutzoberflächen gleich gestalten mit Wiedererkennungsnutzen.

- Grafisch informieren statt sprachlich-textlich.
- Nur jeweils die Aktionsmöglichkeiten auf dem Bildschirm darstellen, die im aktuellen Kontext sinnvoll sind, um Überfrachtung zu vermeiden.

Der allgemeine Bekanntheitsgrad eines Prinzips hilft bei neuartigen, komplexen Aufgabenstellungen, Schulungsaufwand und Verständnisprobleme zu verringern.

Es ist zu beobachten, dass weltweit bekannte, in allen Dienstleistungs- und Industriezweigen eingesetzte Programme der Textverarbeitung, Tabellenkalkulation und grafischen Präsentation zunehmend die gleichen Basiselemente für ihre Nutzoberflächen verwenden. Diese verschiedenen Programme haben daher für Anwender das gleiche „look and feel“. Kennt man eines dieser Programme, kann man viele der Funktionen anderer Programme mit gleichen Elementen sofort ohne Schulung nutzen. Genau diese allgemein bekannten Elemente sollen auch dem Forschungsziel dieser Arbeit dienen, um sicherzustellen, dass der zusätzliche Lernaufwand für die hier konzipierte Nutzoberfläche minimal wird. Dabei wird gleichzeitig untersucht, ob benutzerführende Texte vermieden werden können, um von Landessprachen unabhängig zu bleiben und weltweit ohne Übersetzungen auszukommen.

Der Zweck und das Verhalten folgender Basiselemente von Nutzoberflächen dürfen für Ingenieure weltweit ohne zusätzliche Schulung als bekannt vorausgesetzt werden, wenn sie nach den hier konzipierten Regeln gestaltet sind:

Eingabefeld mit Sinnbild und Drehfeld



Bild 4-1: Sinnbild „Winkel“, Eingabefeld und Drehfeld

Links vor einem Eingabefeld wird ein Sinnbild angeordnet, das unabhängig von Sprachen anzeigt, was der in das Eingabefeld eingegebene Wert bewirkt. Bild 4-1 zeigt das Eingabefeld für die Eingabe eines Winkels. Enthält ein Eingabefeld bereits

einen voreingestellten, selbsterklärenden Wert, z. B. eine national bekannte Werkstoff-bezeichnung wie ST37, ist das vorangehende Erklärungssymbol überflüssig. Der aktuelle Wert im Eingabefeld kann direkt überschrieben oder mit Klick auf eine der Pfeiltasten des Drehfelds in objektspezifischen Inkrementen erhöht bzw. erniedrigt werden. Das Drehfeld verhindert zudem Schreibfehler, wenn nur aus dem Vorrat konkret existierender Möglichkeiten gewählt und nicht geschrieben wird. Mit dem Sinnbild ist zur weiteren Erläuterung eine Kurzinformation und über die F1-Taste eine Online-Hilfe verbunden.

Rechtsklick auf ein Eingabefeld liefert in Corel Draw nicht nur ein Kontextmenü, sondern vergrößert zusätzlich das Eingabefeld so weit, dass auch bei längeren Eingaben alle Zeichen in das Feld passen und nicht teilweise „herausrutschen". Im Kontextmenü ist für dieses Feld das Inkrement einstellbar, um das sich der Wert im Eingabefeld nach Klick auf ▲ erhöht bzw. nach Klick auf ▼erniedrigt.

Eine Eingabe hat sofortige Wirkung auf das Objekt, d. h. die Eingabe ist nicht modal und erfordert keine zusätzliche, abschließende Bestätigung durch eine OK-Schaltfläche.

Listenfeld

Listenfelder, deren Listenelemente bereits selbsterklärende grafische Symbole sind, benötigen nicht einmal ein vorangestelltes Sinnbild, wie das dargestellte Beispiel Bild 4-2 für Stricharten aus Corel Draw zeigt.

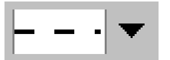


Bild 4-2: Listenfeld

Nach Klick auf den Abwärtspfeil erscheinen die weiteren Elemente der Liste, im Beispiel die Stricharten (Stile) durchgehend, gestrichelt, gepunktet und strichpunktiert. Das gewählte Symbol erscheint im Listenfeld und die Liste schließt sich wieder. Eine entsprechende Lösung ist geeignet für die Wahl einer Maßeinheit (Millimeter, Zoll, Pixel) oder die Wahl aus einer Liste von genormten Strichstärken. Im Werkzeug „Zeichnen" des bekannten Programms Word sind statt eines Listenfeldes Symbole gewählt, die bei Strichstärke, Strichart und Pfeilart den

aktuellen Wert nicht anzeigen, während die aktuellen Farben für Füllflächen, Linien und Text angezeigt werden, siehe Bild 4-3. Erst nach Klick auf die Symbole ohne Anzeige klappt die Liste auf, die dann das aktuelle Element hervorgehoben darstellt. Listenfelder, die den aktuellen Wert ohne zusätzliche Aktion zeigen, sind direkter, also für die Ziele dieser Arbeit geeigneter.

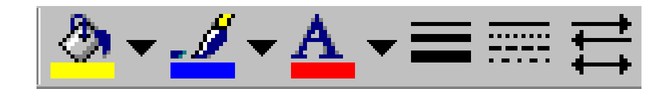


Bild 4-3: Listenfelder im Werkzeug „Zeichnen“ des Programms Word

Die Symbole und Listenfelder des Bildes 4-3 sind in überzeugender Weise textfrei selbsterklärend. Bildlich plastisch wird die Funktion „Fläche mit Farbe füllen“ vor Augen geführt: der umfallende Farbeimer wird eine konturfüllende Farbfläche hinterlassen, aktuell mit der im Rechteck unter dem Eimer gezeigten Farbe. Nach Klick auf den Abwärtspfeil neben dem Farbeimer erscheinen als Liste schachmusterartig weitere Farben zur Wahl. Die gewählte Farbe ersetzt als Bestätigung die bisherige Farbe im Rechteck unter dem Farbeimer.

Das nächste Symbol „Linienfarbe“ verwendet als Sinnbild die Farbe eines Pinselstrichs, die ggf. genau wie zuvor geändert werden kann. Das nächste Symbol „A“ regelt die Farbe eines Textes. Das Symbol mit verschiedenen Strichstärken dient logischerweise zur Wahl der Strichstärke. Das Symbol mit den verschiedenen Stricharten macht die Stricharten (Stile) wählbar, das Symbol mit den Pfeilen bietet die technisch gebräuchlichen Arten von Pfeilspitzen an.

Die drei letzten Symbole zeigen den aktuellen Wert nicht direkt an, sind also keine Listenfelder. Hier sieht man erst nach Wählen des Symbols und Aufklappen der Liste, welches der Listenelemente hervorgehoben ist, also aktuell eingestellt. Ebenso wie bei den ersten drei Symbolen wäre es daher besser gewesen, wenn unterhalb der Symbole für Strichstärke, Linienart und Pfeilart jeweils bildlich der aktuelle Wert dargestellt würde.

Optionsfeld

Sich gegenseitig ausschließende Optionen, z. B. Blechecken ausgerundet, mit Eck-

abschnitt oder eckig, werden mit entsprechender Anzahl Optionsfeldern (Bild 4-4) an grafischen Beispielen dargestellt.

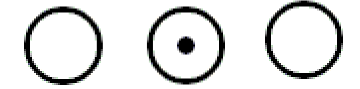


Bild 4-4: Optionsfeld

Die aktuelle, angeschaltete Option ist durch einen Punkt im Kreis markiert. Wählt der CAD-Ingenieur eine andere Option, wird dieses Optionsfeld ein- und das bisherige wie alle anderen ausgeschaltet. Optionsfelder werden deshalb auch „Radiobuttons" genannt, da bei Radios auch immer nur ein Sender gleichzeitig eingestellt werden kann.

Kontrollfeld

Sich gegenseitig nicht ausschließende Optionen werden mit entsprechender Anzahl Kontrollfeldern an Beispielen grafisch dargestellt. Die aktuellen Optionen sind durch einen Haken erkennbar angeschaltet. Wählt der CAD-Ingenieur andere Optionen, muss er diese, sichtbar durch einen Haken im Kästchen, an- und die nicht gewünschten ausschalten.



Bild 4-5: Kontrollfelder

Optionen mit hierarchischer Liste

Unter Menü „Extras/Optionen" bieten CAD-Systeme eine zunächst unüberschaubare Vielzahl an Optionen für das Verhalten und die Voreinstellungen des Systems an. Ohne straffe logische Strukturierung wirkt diese Datenmenge zwangsläufig chaotisch. Langes, oft vergebliches Suchen, wo ein einzustellender Sachverhalt versteckt ist, prägt bei derartigen Systemen die Alltagsarbeit.

Die übliche, einstufig strukturierte Gliederung mit Registerkarten ist mit dieser Aufgabe überfordert. Für die komplexe Baumstruktur logisch aufgebauter Dateisysteme hat sich die Darstellungsweise der hierarchischen Liste in der Praxis am

besten bewährt. Die der Natur der Sache entsprechende logische Strukturierung als mehrstufiger Baum in zusammen gehörende Bereiche macht das zielstrebige Auffinden anhand einer hierarchischen Liste überschaubarer als explizite, tiefe Menüverschachtelungen oder eine Fülle von Registerkarten auf einer Stufe. Bild 4-6 greift daher im linken Fensterbereich die von Dateisystemen bewährte Darstellung logischer Zusammenhänge auch für Optionen auf. Die einzelnen Gruppen von Optionen unterscheiden sich zwar nicht von den üblichen Registerkarten, das Auffinden ist aber wesentlich leichter zielgerichtet. Für eine sprachfreie Gestaltung sind allerdings noch schöpferische Ideen notwendig, um die erläuternden Texte neben den Kontroll- und Optionsfeldern auf den Registerkarten durch selbsterklärende Zeichnungen und Symbole zu ersetzen.

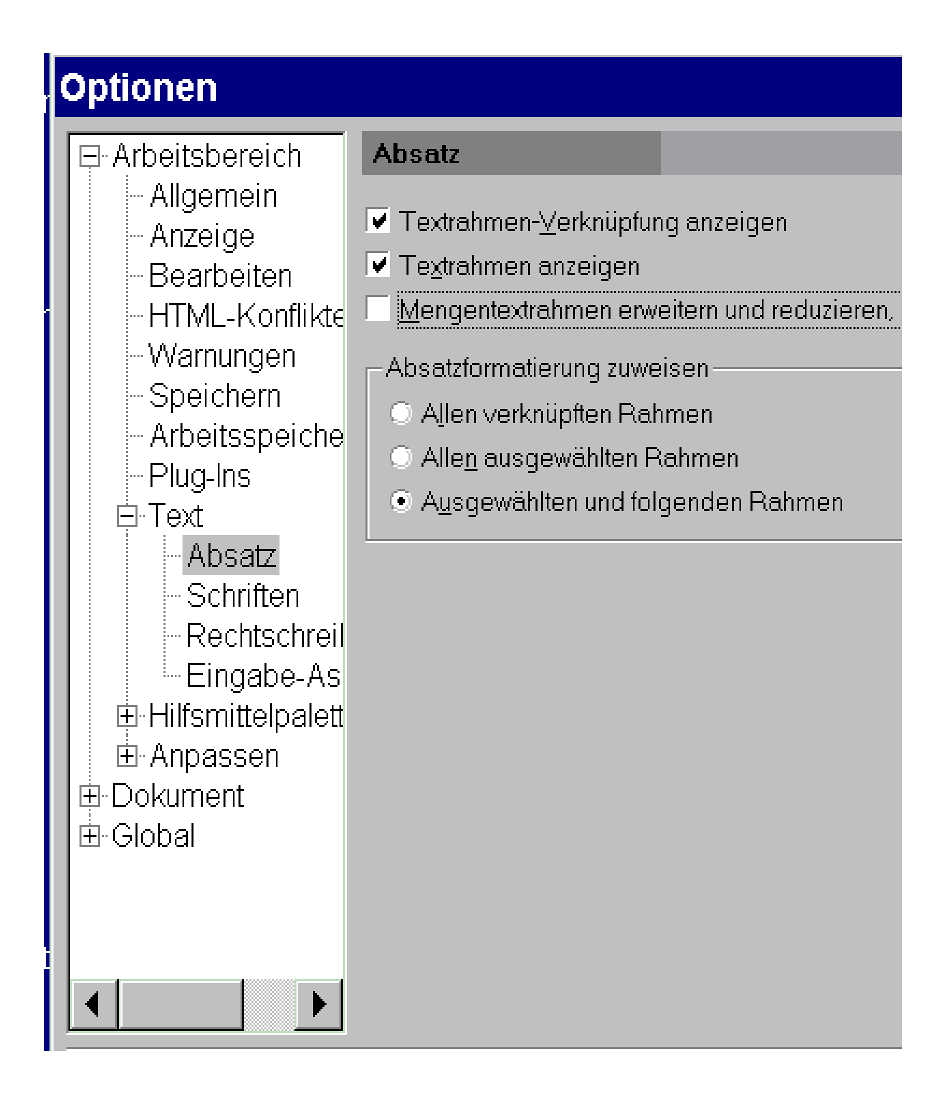


Bild 4-6: „Optionen" mit hierarchischer Liste strukturiert (Quelle: Corel Draw)

Kurzinformation (Quickinfo)

Verweilt der Mauszeiger auf einem Symbol, erscheint bei allen praxisüblichen Programmen ein erläuternder Kurztext. Obwohl Quickinfos somit fundamental gegen die Forderung der Unabhängigkeit von Sprache und Schrift verstoßen, kann ihr Einsatz dennoch vertretbar sein. Eine Quickinfo muss sich nicht aktuell aus mehreren Elementen nach sprachspezifischen, grammatikalischen Regeln aufbauen, sondern wird als eine Einheit gespeichert. Durch programminternen Verweis jeweils auf die landesspezifische Quickinfo-Tabelle müssen diese Einheiten nur einmal übersetzt werden und nicht bei jeder Revision des Programms.

Die gleiche technische Lösung gilt im Übrigen auch für textförmige Leitmeldungen.

Online Hilfe

Steht der Mauszeiger auf einem Symbol, kann bei allen praxisüblichen Programmen über die F1-Taste zu diesem Symbol eine Onlinehilfe auf den Bildschirm geholt werden. Diese Hilfe erläutert mit Texten, Bildern und Schulungsbeispielen die Leistung, die das Symbol bietet und die Bedienung, die es erfordert. Hier liegt es in der Natur der Sache, dass Unabhängigkeit von Sprache und Schrift unmöglich ist. Das Ziel dieser Arbeit wäre in höchster Vollkommenheit erreicht, wenn trotz Verzichts auf sprachgebundene Texte die unmittelbare Verständlichkeit der hier konzipierten Nutzoberfläche so umfassend verwirklicht wird, dass zusätzliche Onlinehilfe tatsächlich weder von Anfängern noch von Experten für Sonderfälle benötigt wird. Selbst einfachste Zeichensysteme mit geringem Leistungsumfang haben dieses Ziel bisher nicht erreichen können. Es ist jedoch zu beobachten, dass in jedem Land von Schulungseinrichtungen Kurse mit landesspezifischen Schulungsunterlagen für CAD-Systeme angeboten werden. Diese Einrichtungen leben von Schulungen und der Erstellung von Schulungsunterlagen. Wenn der Programmentwickler nur Schulungseinrichtungen autorisiert, die die Unterlagen im Programm und im Internet nutzbar als PDF-Dateien mit Hyperlinks gestalten, dann sind diese Unterlagen zugleich als landesspezifische Online Hilfe verwendbar.

Die Integration landesspezifischer Online Hilfe von Partnerfirmen ist daher systematisch und durchgehend einheitlich vom Programmentwickler vorzusehen und bei der Autorisierung von Partnern vertraglich zu beachten.

Eigenschaftsleiste

Mit neuesten Softwarewerkzeugen gestaltete Zeichenprogramme, z. B. Corel Draw (teilweise auch die Werkzeugleiste „Zeichnen“ in Word 2000), zeigen zu jedem aktuell bearbeiteten Objekt eine spezifische Eigenschaftsleiste, in der die Werte der einzelnen Eigenschaften je nach Art numerisch mit Sinnbild oder grafisch angezeigt und ggf. mit sofortiger Wirkung geändert werden können, Bild 4-7.



Bild 4-7: Objektspezifische Eigenschaftsleiste für A4-Format in Corel Draw

Im allgemein bekannten Textverarbeitungsprogramm Word sind in einer Werkzeugleiste für 2D-Grafik neben Werkzeugen zum Erzeugen von Linie, Pfeil, Rechteck oder Ellipse auch deren gemeinsame Eigenschaften einstellbar, z. B. Strichstärke und Linienart, Bild 4-8 (siehe auch Bild 4-3).

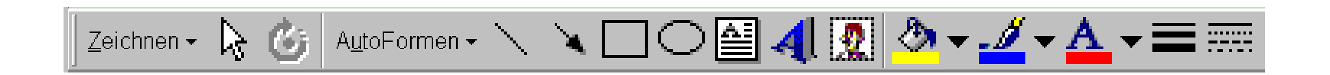


Bild 4-8: Werkzeugleiste „Zeichnen“ mit Eigenschaften Strichstärke und Linienart

Eigenschaftsleisten sind pro Objekt individuell aus den in diesem Abschnitt beschriebenen Elementen mit Sinnbildern, Eingabefeldern, Listenfeldern und Symbolen zusammengestellt. Abgesehen von der sehr beschränkten, kleinformatigen Grafik der Symbole, fehlen Ihnen jedoch anschauliche grafische Darstellungen, also Zeichnungen und Bilder statischer oder gar dynamischer Art. Es ist daher für das Ziel dieser Arbeit wesentlich, die Möglichkeiten der Eigenschaftsleisten mit den Möglichkeiten der Leitbilder und Voreinstellungs-Dialogfelder wie in BOCAD-3D zu kombinieren zu Eigenschaftsfenstern. Diesen umfassenderen Ansatz beschreibt Abschnitt 4.2.9.

Allgemein übliche Funktionsgriffe

In international verbreiteten Programmen wurden zumindest in neueren Versionen einige elementare Funktionsgriffe und Anfasser üblich, z. B. zum Verschieben oder

Kopieren von Objekten. Das Funktionsgriffen und Anfassern zugrunde liegende Prinzip ist besonders geeignet für eine direkte, selbsterklärende Nutzoberfläche hoher Effizienz. Dieses Prinzip nimmt eine besondere Schlüsselstellung ein. Funktionsgriffe werden daher in einem eigenen Abschnitt 4.2.5 behandelt.

4.2.1 MDI (Multiple Dokument Interface) und Mehrfenster-Technik

Unter MDI versteht man eine Oberfläche, mit der mehrere CAD-Systeme gleichzeitig geöffnet werden können. Die MDI Unterstützung wurde von mehreren praxisüblichen Software-Produkten verwirklicht, z. B. AutoCAD, MS Office, StarOffice, PhotoShop usw. Für CAD Software bietet MDI besondere Möglichkeiten, [38], [40]:

- Grafikobjekte aus einer Datei lassen sich in eine andere Datei direkt kopieren.
- Grafikobjekteigenschaften sind übertragbar. Linienstrichart, -rapport und -strichstärke sowie Layereinstellungen werden in die Zieldatei, in die die Grafikobjekte kopiert werden, ebenfalls übertragen.
- Beim Wechsel von einem CAD-Modell zum anderen muss ein aktivierter Befehl nicht abgebrochen werden, sondern kann bei der Rückkehr direkt dort fortgesetzt werden, wo er unterbrochen wurde.
- Es können mehrere Dateien gleichzeitig geöffnet werden, z. B. lassen sich AutoCAD-Dateien in Windows Explorer auswählen, dann können sie über Drag&Drop in AutoCAD 2000 eingefügt werden.

Bild 4-9 zeigt eine MDI-Umgebung in AutoCAD 2000.

Im untersuchten CAD-System können ebenfalls beliebig viele Fenster mit Grafikdateien geöffnet werden, die alle verschiedene Sichten des 3D-Modells darstellen.

Die Tabelle 4-1 zeigt die unterschiedlichen Eigenschaften von Multiple Dokument Interface und Multiple Windows Interface.

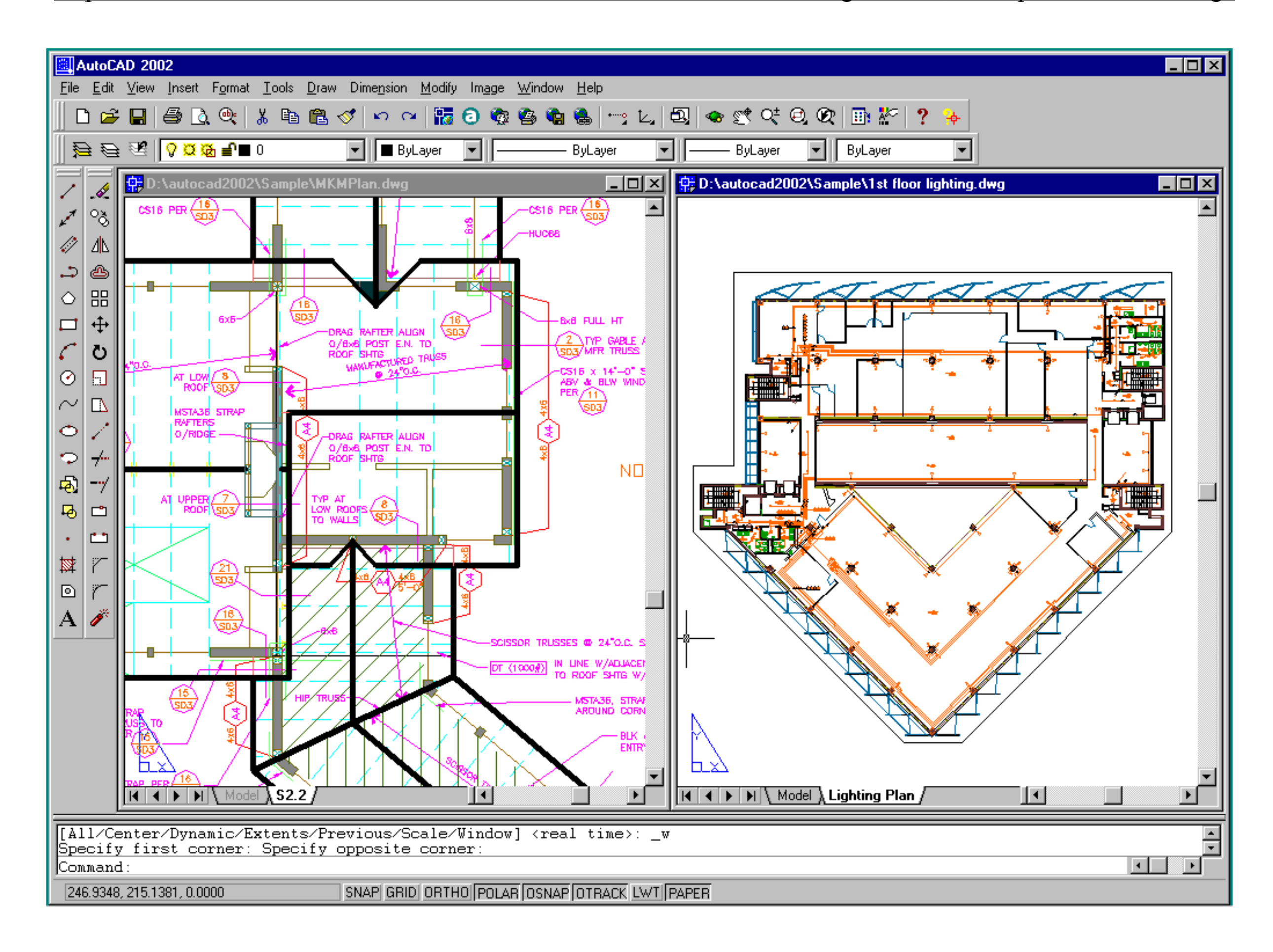


Bild 4-9: MDI: zwei gleichzeitig geöffnete Grafik-Dateien für Drag&Drop

Eigenschaften	*Multiple Dokument Interface*	*Multiple Windows Interface*
Anzahl der zu öffnenden Dateien	Mehrere Dateien können gleichzeitig geöffnet werden.	Nur ein Projekt kann geöffnet werden.
Anzahl der zu öffnenden Fenster	Mehrere Fenster können zwar gleichzeitig geöffnet werden, aber verschiedene Fenster gehören zu verschiedenen Dateien. Manche Systeme teilen ein Ansichtsfenster ein, um mehrere Ansichten darzustellen.	Mehrere Fenster können gleichzeitig geöffnet werden, alle Fenster gehören zu einer gemeinsamen Projektdatenbank und werden aus dieser hergeleitet.
Grafikelemente Kopieren	Grafikelemente können beliebig von einer Datei in eine andere kopiert werden.	Grafikelemente können beliebig von einem Fenster in ein anderes kopiert werden. Alle Fenster stellen das gleiche Modell dar, weil sie Informationen von einer identischen Datenbank herleiten.
Einfluss der Einstellung der Grafikeigenschaften	Die Eigenschaften der kopierten Grafikelemente werden in der Zieldatei übernommen.	Die Eigenschaften der kopierten Grafikobjekte werden ebenfalls kopiert.

Tabelle 4-1: Vergleichen MDI und MWI

Ein typisches Multiple Windows Interface wurde in BOCAD-3D verwirklicht. Diese Leistung kennzeichnet den eigenen Charakter von BOCAD. Bei der Konstruktion von Stahlbauten hat MDI keine Bedeutung, da von einem CAD-Konstrukteur nur ein Projekt bearbeitet wird. Im Vergleich dazu spielt Multiple Windows Interface eine besondere Rolle, wenn Stahlbauteile, z. B. Anschlüsse detailliert werden müssen. Die Möglichkeit, mehrere Fenster mit verschiedenen Blickrichtungen auf den Anschluss gleichzeitig zu öffnen, bietet dem Anwender ideal einfache Ansichten für interaktives Konstruieren. So kann der Benutzer seine Konstruktion am besten kontrollieren, d. h. das Ergebnis seiner interaktiven Konstruktionsschritte überprüfen. Bild 4-10 veranschaulicht ein Beispiel der multiplen Fensteroberfläche von BOCAD-3D.

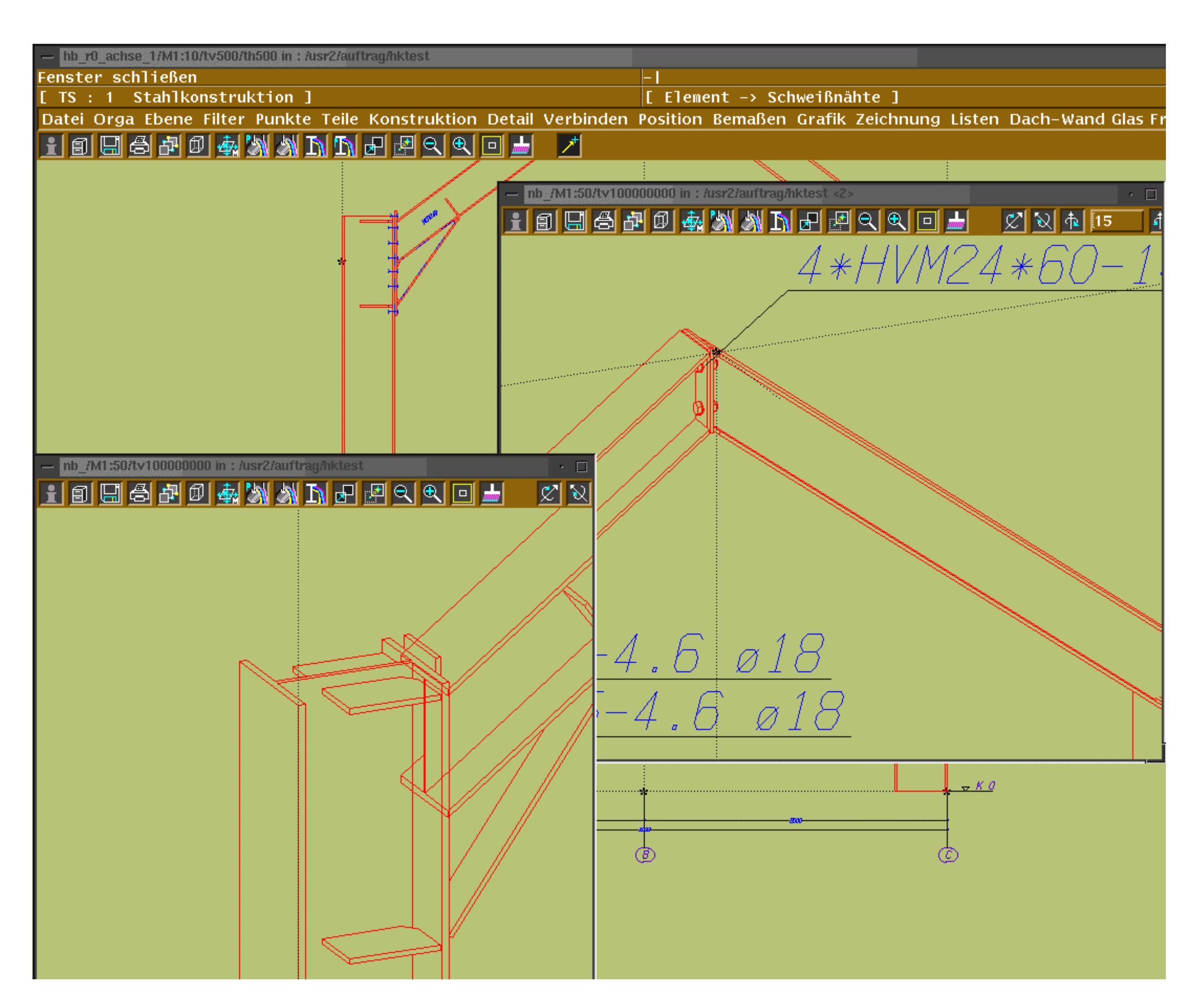


Bild 4-10: Eine Mehr-Fenster-Oberfläche in BOCAD-3D

4.2.2 Bilddarstellung

Ein virtueller Konstruktionsarbeitsraum bietet dem Anwender größte Freiheit der Bilddarstellung. Sie ist eine der am meisten in der Praxisanwendung aufgerufenen Funktionen. Ihre Leistungen haben direkten Einfluss auf die Effizienz des Systems.

Folgende Leistungen der Bilddarstellung muss ein CAD System bieten, um dem Anwender die Konstruktion zu erleichtern und die Effizienz zu erhöhen:

- Echtzeit - Verhalten der Bilddarstellung. Dabei können zwei Möglichkeiten berücksichtigt werden:

  – Nutzung der Hardware:

  Beispielsweise kann man mit dem Rollknopf einer IntelliMaus die Darstellung direkt vergrößern oder verkleinern, ohne zuvor ein Menü oder ein Icon wählen zu müssen.

  – Durch Software zur Analyse von Handbewegungen:

  Beispielsweise kann man durch bestimmte Bewegungen mit Mausfunktionen für die Darstellung aufrufen, z. B. Verschieben des Bildausschnitts.

- Vielseitige ZOOM-Optionen werden benötigt:

  – ZOOM - Echtzeit:

  Die Darstellung kann durch Auf- und Abbewegen der Maus bei gedrückter linker Taste vergrößert bzw. verkleinert werden.

  – ZOOM - Gesamt (Vollbild):

  Alle Objekte werden komplett sichtbar.

  – ZOOM - Mitte:

  Ein beliebiger Punkt in der Zeichnung wird markiert, der anschließend in die Mitte des Darstellungsfensters verschoben wird. In dieser Lage wird die Darstellung mit einem Faktor vergrößert oder verkleinert.

  – ZOOM - Vorher:

  Der vor der aktuellen Darstellung angezeigte ursprüngliche Zeichnungs-ausschnitt wird schnell wieder hergestellt.

– ZOOM - Fenster:

Ein Bildausschnitt kann durch Ziehen eines Rechtecks vergrößert werden.

– ZOOM - Faktor:

Die Bilddarstellung kann um einen festgelegten Faktor vergrößert werden.

- Pan:

  Die Darstellung des virtuellen Bildes wird bei gleichbleibender Größe in Echtzeit im Fenster verschoben, d. h. der dann im Fenster enthaltene Bildausschnitt des virtuellen Bildes wird sichtbar.

- Hilfsgrafik ist auf dem Bildschirm und dem Plotter ein- oder ausschaltbar.
- Übersichtsfenster:

  Das Übersichtsfenster zeigt eine Ansicht der Zeichnung in einem separaten Fenster als Navigationshilfe an. So kann der Anwender schnell über ZOOM oder Pan navigieren, ohne eine Menüoption, ein Icon oder einen Befehl auszuwählen.

- Eine Ansicht der aktuellen Zeichnung kann gespeichert und zu einem späteren Zeitpunkt wieder aufgerufen werden.
- Die Anzeige von Text, Linienstärke, Füllung und Konstruktionspunkten könnte von dem Anwender deaktiviert werden.
- Die Darstellungsgenauigkeit bzw. Darstellungsarten für 3D-Objekte können geändert bzw. gewechselt werden.

4.2.3 Koordinateneingabe und Benutzerkoordinatensystem

Koordinateneingabe und -anzeige

Um eine einzelne Konstruktions- oder Platzierungsfunktion auszuführen, fordert das CAD-System den Anwender meist auf, einen Platzierungspunkt zu wählen. Der Punkt kann durch den Fangmodus von Konstruktions- oder Grafikelementen ermittelt oder frei gepickt werden. Eine andere Möglichkeit kann die direkte Eingabe der Punktkoordinaten sein. Der Benutzer wählt aus, ob diese Koordinaten sich auf das globale Koordinatensystem oder auf das von dem Benutzer selbst definierte Koordinatensystem beziehen sollen. Darüber hinaus sollen die Koordinaten des

Punktes, der durch die aktuelle Mauszeigerposition auf dem Bildschirm festgelegt ist, in der Statuszeile angezeigt werden. Damit kann der Konstrukteur die Position des Mauszeigers leicht kontrollieren. Diese Koordinaten sollen sich auf das aktive Benutzerkoordinatensystem des aktuellen Fensters oder Teilbildes beziehen. Drei unterschiedliche Anzeigearten sind möglich:

- Dynamische Anzeige:

  Koordinaten werden stets aktualisiert, wenn sich der Cursor bewegt.
- Statische Anzeige:

  Koordinaten werden nur aktualisiert, wenn ein Punkt markiert wird.
- Entfernung und Winkel:

  Wenn Grafikobjekte gezeichnet oder Bauteile platziert werden, für die zwei Punkte gewählt wurden, wird der Abstand und der Winkel des Vektors zur lokalen X-Achse angezeigt.

Mit Hot-Keys oder durch Einstellung der Systemvariablen kann der Anwender zwischen den drei Arten der Koordinatenanzeigen wechseln.

Benutzerkoordinatensystem

Für 2D-Grafik sind zwei Koordinatensysteme besonders geeignet: das kartesische und das polare Koordinatensystem. Das kartesische Koordinatensystem hat zwei Achsen: X, Y (und Z im 3D-Raum). Im polaren Koordinatensystem werden Punkte mit Hilfe eines Winkels und eines Abstands definiert. Eine Erweiterung der polaren Koordinaten aus dem 2D-Raum erfolgt durch die Zylinderkoordinate bzw. die Kugelkoordinate im 3D-Raum. Die Eingabe von Zylinderkoordinaten ist ähnlich wie die von 2D-Polarkoordinaten, jedoch wird eine zusätzliche Koordinate auf einer Achse eingegeben, die lotrecht zur XY-Ebene liegt. Im Vergleich dazu wird ein Winkel zur XY-Ebene im Kugelkoodinatensystem statt der Koordinaten in Z-Richtung festgelegt.

Um Grafikelemente oder Bauteile leicht zu platzieren, dient ein Werkzeug, mit dem der Benutzer selbst ein aktuell geeignetes Koordinatensystem definieren kann, im Folgenden BKS genannt. Das Werkzeug soll folgende Leistungen besitzen:

- Innerhalb eines beliebigen Fensters oder Teilbilds können verschiedene BKS

definiert, gespeichert, aufgerufen und gelöscht werden.

- Ein globales Welt-Koordinatensystem WKS, das auf dem kartesischen Koordinatensystem basiert, ist als Voreinstellung eingestellt.
- Der Benutzer kann festlegen, ob und wie das BKS-Symbol für das Koordinatensystem in den verschiedenen Ansichtsfenstern angezeigt wird.
- Durch folgende Methoden ist der Benutzer in der Lage, ein neues BKS zu definieren:

  – Neue XY-Ebene definieren

  – Neuen Ursprung definieren

  – BKS nach einem bestehenden Objekt ausrichten

  – BKS nach der Fläche eines Körpers ausrichten

  – BKS an der aktuellen Ansichtsrichtung ausrichten

  – BKS um eine der eigenen Achsen drehen

  – BKS zu einer festgelegten Z-Achse erstellen

  – BKS aus einer Liste wählen, die das CAD-System vorgibt

In CAD-Bausoftware ist ein Gebäuderaster üblich, das wie ein globales WKS funktioniert. Mit Hilfe des Rasters kann der Anwender Punkte definieren und dann Grafikelemente und Bauteile sowie Teileverbindungen exakt platzieren.

4.2.4 Exaktes Platzieren ohne Dateneingabe, Objektfang, Einrasten und Spurvektor

Eines der häufigsten und aufwendigsten Probleme in der Alltagsarbeit des rechnergestützten Konstruierens ist das exakte Platzieren von Objekten bei ihrer Erzeugung oder Änderung. Exakte Lagedaten zu ermitteln und einzugeben wäre der höchste Arbeitsanteil der interaktiven Arbeit mit dreidimensional arbeitenden CAD-Systemen, wenn nicht ebenso elegante wie intelligente Lösungen für dieses Problem gefunden werden.

3D-Systeme der oberen Leistungsklasse gestatten, Objekte gleichzeitig in mehreren Fenstern mit unterschiedlichen, beliebig wählbaren Betrachtungsrichtungen darzustellen und in allen diesen Fenstern zu bearbeiten. Wie im Abschnitt 4.2.1

dargestellt, CAD-Ingenieure wählen dabei zur Vereinfachung der Arbeit am Bildschirm vorzugsweise mindestens eine Ansicht, in der das bearbeitete Objekt in tafelparalleler Lage zum Bildschirm liegt. Stahlbau-Objekte wie Profilstäbe oder Bleche, Schrauben und Schweißnähte sind dann in einer Hauptansicht dargestellt, in der Ebenen oder Kanten des Objekts ebenfalls tafelparallel sind und sich somit unverzerrt abbilden. Für exaktes Platzieren weiterer Objekte oder das Detaillieren von Objekten ist der Kontakt zu vorhandenen Objekten, ihren Ecken und Kanten die entscheidende geometrische Forderung. Bauobjekte schließen naturgemäß aneinander an, man verbindet sie durch Anschlüsse. Selbst für das kollisionsfreie Platzieren von ergänzenden 2D-Grafiken und Texten gelten die gleichen geometrischen Forderungen.

Fangmechanismen sind in allen CAD-Systemen ein wesentliches Gestaltungs-element der Nutzoberfläche, um prinzipiell überflüssige quantitative Eingaben und Pickaktionen zu vermeiden. Auch Standardsoftware der Textverarbeitung, Tabellenkalkulation, Präsentation und Flussdiagramm-Erstellung nutzt konsequent die Möglichkeiten der Eingabe- und Aktionsreduzierung durch Fangmechanismen. Speziell in CAD-Systemen erübrigen Fangmechanismen oft Messungen im 3D-Modell, Eingabe genauer Zahlenwerte, handwerklich präzises Platzieren oder die Konstruktion von zusätzlichen Punkten im Raum.

Diese naturgegebenen, funktionsbedingten Lagerestriktionen lassen sich elegant nutzen, um die aufwändige Eingabe von Lagekoordinaten einzusparen, also das Ziel „eine einfache, effiziente und anschauliche Nutzoberfläche“ zu erreichen. Dazu ein Beispiel:

Schiebt man am Bildschirm eine Stütze in die Nähe einer senkrechten Rasterlinie, ist es außerordentlich hilfreich, wenn die Stütze mit ihrer Außenkante ohne weitere Maßangabe automatisch bündig an der Rasterlinie einrastet. Schiebt man die Stütze weiter, sollte sie automatisch mit ihrer Mittellinie bzw. Schwerachse an der Rasterlinie einrasten. Diese funktionalen Lagen einer Stütze besitzen die höchste Gebrauchshäufigkeit. Es vereinfacht und beschleunigt das Arbeiten, wenn der CAD-Konstrukteur durch dieses intelligente Verhalten des Objekts „Stütze“ mit einer direkten Mausbewegung am Verschiebegriff des Objekts nur qualitativ andeuten

muss, was er meint. Die quantitativ exakte Platzierung der Stütze erfolgt rechnerintern durch hochgenaue Analyse der räumlichen Nachbarschaft der beteiligten Objekte und der angedeuteten Verschieberichtung.

Um auch schnell und präzise 2D-Grafik zeichnen zu können, sollen Werkzeuge zur Verfolgung und zum Fangen der Ankerpunkte von 2D-Grafik zur Verfügung stehen. Weil 2D-Werkzeuge häufig in der Praxis benutzt werden, sollten ihre Fangmodi in einer Icon-Liste am Bildschirmrand angeordnet werden, damit der Anwender schnell den Modus wechseln kann.

Folgende Merkmale werden in der Praxis gefordert:

- Fangwinkel und Basispunkt änderbar.
- Punktfangmodi sowohl einmalig als auch fortlaufend.
- Fangmodus entlang des Winkels bei Polarfang eingestellt.
- In 3D-Darstellung werden Fangabstände und Fadenkreuz an den entsprechenden Achsen oder einer Ebene ausgerichtet.
- Ein Hilfsmittel für Verfolgen und Punktfilter einstellbar, [38].

Tabelle 4-2 veranschaulicht die praxisüblichen Punktfangmodi.

Punktfangmodus	***Beschreibung***
Endpunkt	Es wird der Endpunkt einer Linie oder eines Kreisbogens gefangen.
Schnittpunkt	Es wird der Schnittpunkt zweier Linien, einer Linie und eines Kreises oder Kreisbogens, zweier Kreise oder zweier Kreisbögen, einer Linie und einer Ebene gefangen.
Mittelpunkt	Es wird der Mittelpunkt einer Linie oder eines Kreisbogens gefangen.
Zentrum	Es wird das Zentrum eines Kreises oder Kreisbogens, einer Ellipse, eines Rechtecks, eines regelmäßigen Polygons gefangen.
Quadrant	Es wird der nächstgelegene Quadrant von Bögen, Kreisen und Ellipsen gefangen.
Basispunkt	Es wird der Einfügepunkt von Texten oder Symbolen gefangen.
Punkt	Es werden bereits existierende Punkte gefangen.
Lot	Es werden die Punkte im Objekt, die eine lotrechte Ausrichtung bilden, gefangen.
Parallele	Es wird der Punkt auf einer Geraden markiert, der parallel zum ausgewählten Objekt verläuft.
Tangente	Es wird der Punkt auf einem Kreis oder Kreisbogen gefangen, der durch die Berührung einer Geraden erzeugt wird.
Nächster	Es wird der Punkt markiert, der dem Auswahlpunkt am nächsten liegt.
Teile	Es werden die Eckpunkte, die Wurzelmaßpunkte sowie der Schwerpunkt eines Teiles gefangen. (für Stahlbau CAD-System)
Schrauben	Es werden Punkte aus den Schraubenpolygonen, Schraubenachsen und -mittelpunkte gefangen. (für Stahlbau CAD-System)
Naht	Es werden Punkte in den Nahtpolygonpunkten gefangen. (für Stahlbau CAD-System)

Tabelle 4-2: Punktfangmodi, [38]

AutoCAD bietet einen entsprechenden Werkzeugkasten „Objektfang“ (Bild 4-11), mit dem die Kombination der geometrisch markanten Punkte von Objekten wählbar ist, die als Marker entsprechender Form erscheinen sollen. Der Objektfangmodus ist einstellbar (Bild 4-12).

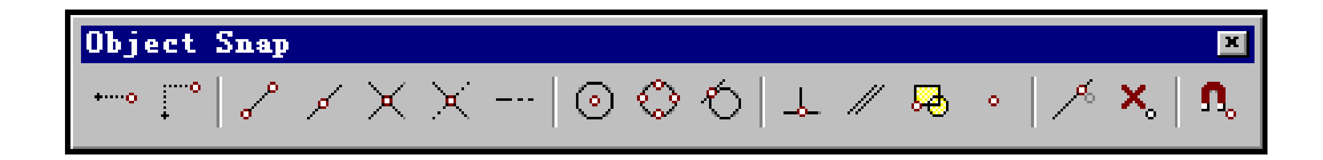


Bild 4-11: Werkzeugkasten Objektfang (Quelle AutoCAD 2000)

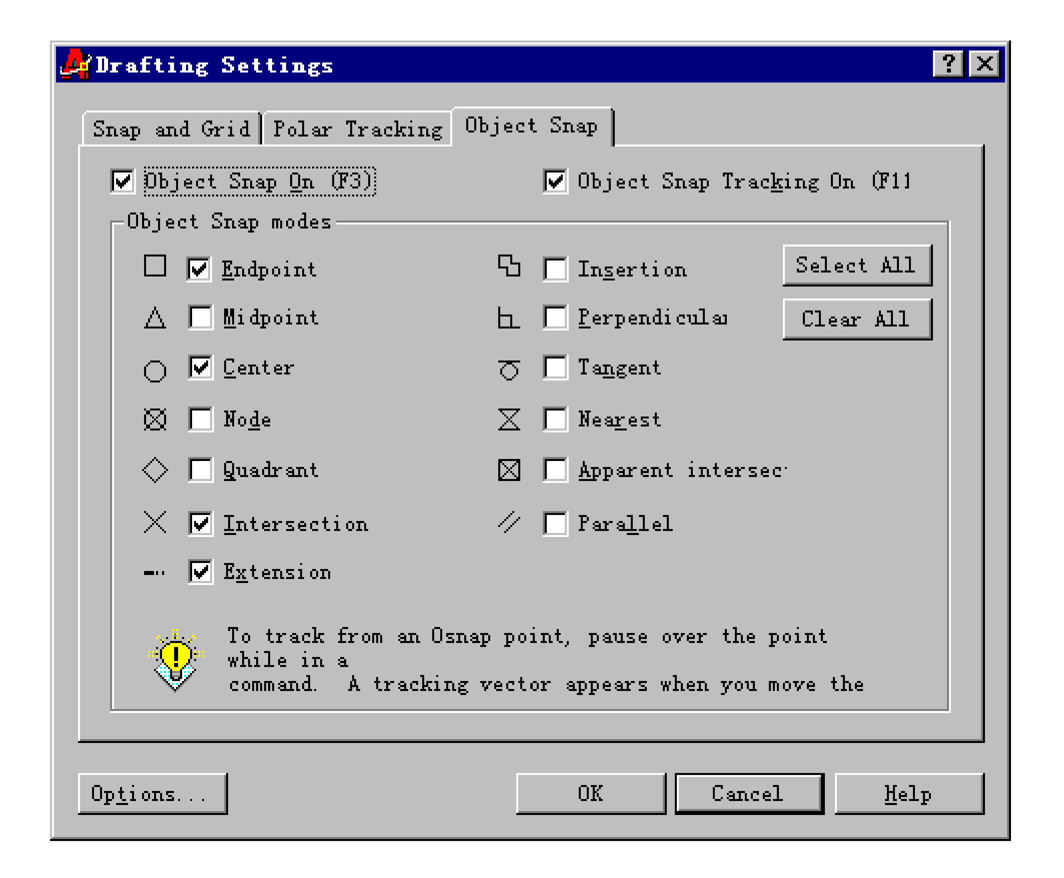


Bild 4-12: Einstellungsdialogbox Objektfang (Quelle AutoCAD 2000)

Ein wirksames Werkzeug zur Reduzierung des Eingabeaufwands auf das absolut notwendige Maß sind daher intelligente Markersymbole, die sich automatisch an Objekten in der Nähe des Mauszeigers anbieten und den Mauszeiger bei weiterer Annäherung „magnetisch" anziehen bis er exakt am Punkt des Markers einrastet, siehe auch Magnetsymbol in Bild 4-11. Die Lage des Markers auf einem Linienende, einer Linienmitte, einem Linienschnittpunkt oder der Schraubenachse bestimmt das spezifische Markersymbol, damit eindeutig erkennbar wird, worauf der Mauszeiger eingerastet ist. Der getroffene Marker leuchtet dazu vergrößert auf.

Die Gefahr, bei nicht ausreichender Vergrößerung (Zooming) versehentlich ein gar nicht gewünschtes Element zu picken und dadurch falsche Abmessungen mit einer Kette von Folgefehlern zu erzeugen, steigt allerdings bei hoher Dichte sich anbietender Marker. Bei einem dünnen Steg, den man in Stegmitte picken wollte, kann aus Versehen die benachbarte rechte oder linke Stegflanke getroffen werden. Achtet man nicht darauf, welcher Marker aufleuchtet, bleibt in diesem Fall der Fehler so klein, dass er zumindest auf dem Bildschirm kaum offensichtlich wird. Bei der automatischen Ermittlung von Gleichteilen werden sich aber später nicht erwartete Unterschiede ergeben oder im schlechtesten Fall Montageprobleme.

Es wird daher vorgeschlagen, die magnetische Anziehungskraft von Markern in Anzahl Pixel je nach Bedeutung unterschiedlich groß zu gestalten, d. h. die „magnetische Anziehung“ des Mauszeigers durch einen Marker intelligent variabel einzustellen. Systempunkte des Gebäuderasters sind dann wirksamer einzustellen als alle sonstigen Marker. Aus diesem Grunde wird im CAD-System BOCAD-3D vorrangig mit Punkten gearbeitet.

Die Betriebssicherheit von Markern kann erhöht werden, wenn Marker den Mauszeiger kontextabhängig je nach Wahrscheinlichkeit unterschiedlich stark anziehen und der Mauszeiger auf dem annähernd getroffenen Punkt einrastet. Zur visuellen Rückmeldung ändert er seine Gestalt in die Form des Markers. Die Stärke des „magnetischen“ Anziehens kann für jede einzelne Art der Marker am sinnfälligsten über die Größe des Fangradius in Pixel eingestellt werden. Bei Häufungen, in denen die Fangkreise der einzelnen Marker stark überlappen, ist aus Sicherheitsgründen Zooming vorzusehen.

Analoge Fangmechanismen bis hin zu ästhetisch befriedigendem Ausrichten von Textangaben auf Zeichnungen reduzieren und vereinfachen den interaktiven Arbeitsaufwand des CAD-Konstrukteurs bei allen Operationen mit Objekten wesentlich. In Kapitel 5 wird deshalb zu den einzelnen Objekten des Stahlbaus jeweils mit durchdacht und konzipiert, welche Fangmechanismen eine intelligente Hilfe darstellen.

Einrastender Spurvektor

Sehr anschaulich, direkt und unmittelbar verständlich ist bei fast allen gestaltverändernden oder gestalterzeugenden Funktionen ein Spurvektor, der gummibandartig der Mausbewegung folgt. Ein Spurvektor ist also quasi ein dehnbarer Griff, der anschaulich durch seine Länge ein aktuelles Maß und eine Richtung anzeigt. Als Eigenschaftsfenster wird dazu folgende Anordnung vorgeschlagen, die Bestandteil aller Eigenschaftsfenster der Objekte sein soll, die mit Hilfe eines Spurvektors anschaulich bearbeitet werden können.

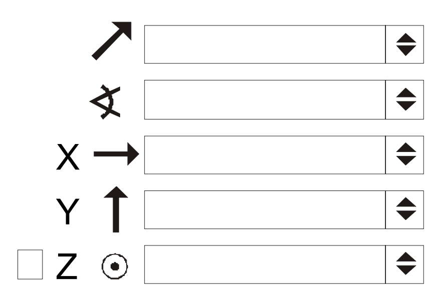


Bild 4-13: Eigenschaftsfenster „Spurvektor“

Das Eigenschaftsfenster (Bild 4-13) zeigt zunächst nach Klick auf den Marker des Objekts im Feld „ ↗ “ die im zuletzt genutzten gleichartigen Befehl verwendete Länge des Spurvektors als Voreinstellung an. Bestätigung mit der Eingabetaste führt den aktuellen Befehl ohne weitere Mausbewegung mit exakt dieser Länge aus. Statt nur eines Wertes kann hier auch eine Liste von Werten eingegeben werden, um gerasterte Wiederholungen zu erzeugen. Wird statt Bestätigung durch die Eingabetaste die Maus weitergezogen, zeigt das Eingabefeld „ ↗ “ mitlaufend die aktuelle Länge des Spurvektors vom Marker bis zum Mauszeiger, das Winkelfeld den Winkel zur lokalen X-Achse des Arbeitsfensters, das X-Feld die X-Komponente der Vektorlänge und das Y-Feld die Y-Komponente. Bei einigen räumlich wirkenden Befehlen kann es je nach Fall zweckmäßig sein, die Z-Komponente senkrecht zum Arbeitsfenster zu Null zu setzen oder zu berücksichtigen. Dazu dient das Kontrollfeld vor der Z-Komponente.

Die vorgeschlagene Funktionsweise des Spurvektors sei konkret am Beispiel „Erzeugen und Platzieren eines 2D-Kreisobjekts“ erläutert:

Zum Zeichnen eines Kreises wird zunächst durch Linksklick der Kreismittelpunkt gewählt. Bei weiterem Ziehen des Mauszeigers erscheint eine gummibandartige Linie, der Spurvektor, als Radius vom Mittelpunkt bis zur aktuellen Position des Mauszeigers sowie dynamisch der entsprechende Kreis. Die aktuelle Länge dieses Spurvektors wird mitlaufend im Eigenschaftsfenster des Objekts angezeigt. Dabei ändert sich der Wert für Länge und Winkel des Spurvektors nicht kontinuierlich, sondern in technisch sinnvollen inkrementellen Schritten, z. B. 5 mm wahre Größe und 15 Grad. Diese Eigenschaft ist als „Schritteinheit Spurvektorlänge“ bzw. „Schritteinheit Spurvektorwinkel“ global einstellbar (Wert und Maßeinheit). Nähert sich der Mauszeiger einem Element, das als Objektfang aktiv ist, z. B. Linienende, Ecke, Linienmitte oder Linienschnitt, so wird der Mauszeiger und somit der Spurvektor vom Marker des Elements „magnetisch“ angezogen und rastet dort exakt ein, d. h. ohne Berücksichtigung der inkrementellen Längenschritte, die nur außerhalb der Markerbereiche relevant sind. Der entsprechende Objektfang-Marker leuchtet auf, damit auch bei eng benachbarten Markern sicher zu erkennen ist, welcher Marker gefangen hat. Wird der Mauszeiger über den Wirkradius des Objektfangs hinaus weiter gezogen, erhöht sich die Länge des Spurvektors wieder inkrementell. Nach Beenden des Ziehens am Spurvektor durch Loslassen der linken Maustaste ist im Eigenschaftsfenster eines jeden Objekts der Wert der Länge und des Winkels bei Bedarf, z. B. vermindert um ein Toleranzmaß, zu überschreiben. Das Überschreiben der Spurvektorlänge führt unmittelbar zur entsprechenden Korrektur des Objekts, im Beispiel also des Kreisradius bzw. des dargestellten Kreises.

Somit ist das Schlagen exakter Kreise reduziert auf eine einzige, gezielte Handbewegung: Linksklick auf den exakten Mittelpunkt und Ziehen des Radius als Spurvektor bis er am gewünschten Maß oder Marker einrastet.

Die geschilderte Spurvektor-Methode vereinfacht durch ihr intelligentes Verhalten die Nutzoberfläche nicht nur für das Kreisobjekt, sondern auch für alle übrigen Objekte sinngemäß zur offensichtlich effizientesten Form, einer einzigen Handbewegung mit Mausklick.

Wie alle Befehle wird der laufende Befehl fortgesetzt durch Wahl des nächsten

gleichartigen Objekts oder beendet durch Wahl eines anderen Befehls. Ein Abbruch und Zurücksetzten auf den Zustand vor Beginn des Befehls ist stets durch Taste <Esc> möglich.

4.2.5 Funktionsgriffe

Das Prinzip der direkten, von Landessprachen unabhängigen, Nutzeroberfläche mit Griffen findet sich inzwischen ansatzweise auch in elementaren Textverarbeitungsprogrammen. In „Word 2000 für Windows“ werden zur Größenänderung von Bildern im Text nicht nur Menüs und Symbole angeboten, sondern in neueren Versionen alternativ auch „Griffe“. Nach Klick auf ein im Text eingefügtes Bild bietet das Bildobjekt ein umschreibendes Rechteck an, in dessen Ecken kleine Rechtecke als „Griffe“ erscheinen. Greift man einen der Griffe mit der Maustaste, lässt sich das Bild durch Ziehen mit der Maustaste vergrößern bzw. verkleinern und so der Platzsituation im Text anpassen. Dabei ändert der Mauszeiger seine Form, um die angebotene Leistung zu veranschaulichen.

Über einen weiteren Griff „ “, lässt sich das Bild auf der Textseite verschieben. Das Bildobjekt bietet also selbständig passende Griffe für objektspezifische Modifikationsmöglichkeiten an und reagiert auf entsprechende Aktionen des Nutzers „intelligent“ durch entsprechende Größenänderung oder Verschiebung.

Funktionsgriffe führen ohne Wahl aus einer Befehlsliste mit einer einzigen Aktion unmittelbar am Objekt direkt und anschaulich zum Ziel. Sie erscheinen abhängig von der Objektart (Klasse) direkt an geeigneten Objektstellen. Sie ersetzen als überlegene Technik in wichtigen Fällen Menüs oder gar mehrfach verschachtelte Untermenüs. Da das Griffsymbol direkt an der Aktionsstelle des Objekts erscheint, entfällt die umständliche Mausbewegung zum Bildrand, die Suche und der Klick auf das gewünschte Symbol und die Mausbewegung wieder zurück zum Objekt. Selbsterklärend gestaltete und allgemein bekannte Griff sind sprachfrei, d. h. die Übersetzung in alle Sprachen entfällt ebenso wie der sonst bei Neuerungen erforderliche Dokumentationsaufwand. Durch marktgängige Programme der Textverarbeitung und Tabellenkalkulation ist das Arbeitsprinzip von Funktionsgriffen allgemein bekannt, ebenso die typischen Reaktionen intelligenter Objekte auf

Aktionen mit Griffen.

Die folgende Tabelle zeigt eine Auswahl bekannter oder selbsterklärender Griffe und die von ihnen jeweils erwartete Funktionalität.

Griffe	Funktionalität
X	Löschen des gewählten Objekt, „durchstreichen“
↔	Verschieben in horizontaler Pfeilrichtung (aus Wingdings 3)
↕	Verschieben in vertikaler Pfeilrichtung (aus Wingdings 3)
✥	Verschieben in beliebiger Richtung
◂‖▸	Kante oder Linie parallel verschieben
✥ + <Strg>	Kopieren in beliebiger Richtung
◢ | ◣	Spiegeln um eine Linie (aus Webdings ähnlich AutoCAD)
⚓	Ankerpunkt verschieben
⚓ + <Strg>	Ankerpunkt kopieren (einen zusätzlichen Ankerpunkt erzeugen)
☑	An- bzw. Abschalten (aus Windings)
⊙	Option ausschließend auswählen (aus Windings 2)
🖫	Speichern (aus Windings)
🖫 + <Strg>	Speichern als ..., d. h. Kopie unter neuem Namen speichern
🖶	Drucken bzw. Plotten (aus Webdings)
🗨	Bemerkung hinzufügen (aus Webdings)
?	Onlinehilfe anbieten (aus Webdings)
⃠	Aktion nicht zulässig (aus Webdings)
▲ ▼ ⬍	Wert inkrementell erhöhen, erniedrigen oder beide Möglichkeiten
▶	Untermenü aufklappen (aus Marlett)
Befehl ...	Dialogfeld zum Befehl aufklappen
⏮ ◀ ▶ ⏭	Zum Listenanfang, ein Element zurück / weiter, zum Listen Ende

Tabelle 4-3: Griffe und ihre Funktionalität

4.2.6 Freihandsymbole

Gerade Aktionen mit sehr großer Häufigkeit sollten vorzugsweise in weniger als einer Sekunde ausführbar sein, um deutlich unterhalb der Schwelle der Frustrationstoleranz von CAD-Konstrukteuren zu bleiben. Durch direkte Freihand-bewegungen mit der Maus ist dieses Ziel zuverlässig erreichbar.

Die häufigste Aktion im Bauwesen beim interaktiven Konstruieren am Bildschirm, auch bei Doppelbildschirmen, ist die Ausschnittvergrößerung, Zoomen genannt. Viele CAD-Systeme sehen hierfür am Rand des Bildschirms das Symbol einer Lupe mit Pluszeichen vor. Man muss dann die Maus dorthin bewegen, die Lupe wählen, die Maus zurück bewegen zur unteren linken Ecke der Ausschnittdiagonalen, die linke Maustaste drücken und bis zur rechten oberen Ecke des Ausschnitts ziehen. Um später wieder das ursprüngliche Bild zu bekommen, muss wieder am Bildschirmrand das Lupensymbol gewählt werden, das durch ein Minuszeichen kenntlich gemacht ist. Es leuchtet unmittelbar ein, dass diese Folge von Aktionen ineffizient ist.

Im CAD-System dieser Arbeit erfolgt daher das Zoomen mit einer Handbewegung (bei gedrückter mittlerer Maustaste) direkt längs der Ausschnittdiagonalen. Für das ebenso häufige Zurückblättern in das ursprüngliche, nicht „gezoomte“ Bild genügt ein Klick auf die mittlere Maustaste. Diese Aktionen sind rekursiv, d. h. eine Ausschnittvergrößerung in einer Ausschnittvergrößerung, auch die Schritte zurück, sind mehrfach geschachtelt möglich.

Entartet die diagonale Handbewegung zu einer nahezu waagerechten oder nahezu senkrechten Linie, wird der im Fenster sichtbare Bildausschnitt entsprechend der Handbewegung verschoben. Für diese Pan (wie Panorama) genannte Aktion sind in den meisten CAD-Systemen aufwändigere Verfahren üblich, z. B. horizontale und vertikale Bildlaufleisten, die nicht nur ständig den verfügbaren Platz auf dem Bildschirm unangemessen einschränken, sondern auch Leerwege zum Fensterrand erfordern.

Die gleichen, sehr einfachen und logischen Handbewegungen lassen sich je nach Maustaste, die gedrückt wird, für unterschiedliche Zwecke nutzen. Eine diagonale Handbewegung von unten links nach oben rechts bei gedrückter linker Maustaste

beispielsweise spannt ein rechteckiges „Lasso“ auf, das alle wählbaren Objekte identifiziert, die völlig innerhalb des Lassos liegen. Die gleiche Handbewegung, jedoch von oben rechts nach unten links, identifiziert alle wählbaren Objekte, die ganz oder teilweise innerhalb des Lassos liegen. Mit der rechten Maustaste zeigt diese Handbewegung die Griffe der im Lasso liegenden, wählbaren Objekte.

Linksklick ohne Bewegung, allein auf einen Punkt, identifiziert nur ein wählbares Objekt.

Rechtsklick ins Leere zeigt als Eigenschaftsfenster die Attribute und Voreinstellungen der laufenden Aktion. Rechtsklick auf ein Objekt macht alle funktionsspezifischen Griffe des Objekts sichtbar und wählbar. Zusätzlich erscheint das Eigenschaftsfenster des Objekts.

4.2.7 Objektauswahl, Filtern

Eine besonders häufige Aktion ist das Identifizieren von Objekten einer Klasse, z. B. Bauteilen, Schrauben oder Grafiken, um anschließend eine Methode auf die identifizierten Objekte wirken zu lassen. Häufige und selbst seltenere, komplexe Identifizierungsaufgaben dürfen daher nur Minimalaufwand des Konstrukteurs am Bildschirm erfordern. CAD-Systeme, die prinzipiell immer erst die gewünschte Aktion fordern und dann dazu - über Leitmeldungen dieser Aktion gesteuert - die Objekte identifizieren lassen, benötigen kein Symbol für das Identifizieren. Systeme, die umgekehrt erst die Objekte identifizieren und dann die Aktion wählen lassen, müssen demgegenüber in der Werkzeugleiste ein spezielles Symbol für das Identifizieren von Objekten anbieten. Bild 4-14 zeigt dieses Symbol in der nur leicht unterschiedlichen Gestaltung von Corel Draw und Word.

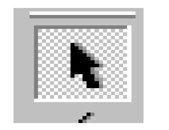
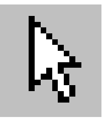
 

Bild 4-14: Symbolformen „Objektauswahl“, „Objekte markieren“

Problemabhängig sind für das Identifizieren mehrere Vorgehensweisen vorzusehen:

Ein einzelnes Objekt wird üblicherweise direkt durch Klick mit der linken Maustaste auf eine Objektdarstellung identifiziert (Linksklick). Eine Liste mehrerer Objekte wird durch Linksklick nacheinander auf die Objekte gebildet. Bei vielen gebräuchlichen Software-Paketen ist dabei gleichzeitig die Schift-Taste < ⇑ > zu drücken, um das Bilden einer Liste zu signalisieren. Bei anderen Systemen genügt es, die Liste nach dem letzten identifizierten Element lediglich durch die Eingabetaste < ↵ > abzuschließen. Bei Listen ist bereits das Irrtumsproblem effizient zu lösen, wenn ein Element irrtümlich in die Liste aufgenommen wurde. Statt die gesamte Liste zu löschen und nochmals von vorn mit der Identifizierung zu beginnen, sehen daher viele Systeme vor, das unerwünschte Element nochmals zu identifizieren und so aus der Liste herauszunehmen. Sinnvoll ist auch, dies nur bewusst durch gleichzeitig gedrückte Taste <Strg> zuzulassen.

Identifizierte Objekte werden zur visuellen Bestätigung kräftiger dargestellt, leuchten also auf. Bei Einzelteil-Objekten ist dies banal, Hauptpositions-Objekte oder Baugruppen jedoch erfordern bereits bauspezifische Intelligenz, denn bei derartigen Objekten muss nicht nur das direkt identifizierte Element aufleuchten, sondern alle über die Objektlogik mit diesem Element verknüpften Objekte. Wenn also ein Element eines Hauptpositions-Objekts identifiziert wurde, muss das Leitteil aufleuchten und alle mit Werkstattverbindungen direkt oder indirekt am Leitteil angeschlossenen Anbauteile, ggf. einschließlich Schraub- und Schweißverbindungen.

Benachbarte Objekte sind effizienter mit einer diagonalen Ziehbewegung der Maus bei gedrückter linker Maustaste über ein ,,Rechtecklasso“ einzufangen, das durch die Diagonale gebildet wird. Alle Elemente der Objektklasse, die vollständig innerhalb des Lassos liegen, bilden die Liste. Unerwünschte Elemente werden bei gedrückter Taste <Strg> direkt abgewählt. Als Variante der diagonalen Ziehbewegung von rechts unten nach links oben werden bei entgegengesetzter Ziehrichtung auch alle Objekte der Klasse in die Liste aufgenommen, die nur teilweise im Lasso liegen. Weiterhin ist eine Technik denkbar, mit der Maus die gewünschten Objekte auf einer umschließenden Kontur zu umfahren.

Auswahllisten, z. B. Material- oder Profiltabellen mit den Namen der verfügbaren

Objekte, gestatten eine Wahl ohne Eingabe des Namens. Da lediglich der Name zu wählen, nicht jedoch einzugeben ist, entfällt der Schreibaufwand und Eingabefehler sind verfahrensimmanent ausgeschlossen. Bei sehr langen Listen werden zur Beschleunigung der Wahl nicht nur Bildlaufleisten angeboten, sondern eine kompakte Vorauswahl nach Eingabe des Anfangsbuchstabens. Profiltabellen oder Schraubentabellen, die jeweils einige tausend Namen umfassen, werden durch diese Technik erst sinnvoll handhabbar.

Ein besonders leistungsfähiges, aber auch methodisch anspruchsvolles Werkzeug zur Identifizierung ist das Bilden einer Liste durch Filter mit situations- oder aufgabenabhängigen Filterkriterien. Das automatische Sammeln der Werkstattzeichnungen aller Stützen eines Gebäudes auf Sammelblättern über die Benennung „Stütze“ ist ein typisches Beispiel. Für diese Leistungen, die völlig individuelles, firmenspezifisches Filtern von Objekten für spezielle Listen gestatten, ist z. B. im CAD-System BOCAD-3D ein Hauptmenü „Filter“ verwirklicht.

4.2.8 Eigenschaftsfenster

Beim Programmieren in der objektbasierten Sprache Visual Basic Application (VBA) fällt angenehm auf, dass zu jedem Objekt, mit dem man z. B. ein eigenes Dialogfeld gestaltet, ein Fenster mit allen Eigenschaften des Objekts angeboten wird. In diesem Fenster sind pro Eigenschaft neben dem aktuellen Wert ggf. als Listenfelder auch alle alternativen Werte angezeigt. Information, Dokumentation (auch über Onlinehilfe) und Eingabe von Werten sind also in einem objektspezifischen Fenster zusammengefasst. Das Objekt reagiert sofort anschaulich auf die Änderung einer seiner Eigenschaften im Eigenschaftsfenster, ohne dass zuvor eine Schaltfläche OK oder Übernehmen gewählt werden muss. Dieses zweckmäßige Fenster für Eigenschaften von Objekten lässt sich gedanklich auf CAD-Systeme übertragen und für die Ziele dieser Arbeit entscheidend verbessern. Dem Eigenschaftsfenster in VBA fehlen nämlich die Vorteile grafischer Darstellung.

Das deutsche Sprichwort „Ein Bild sagt mehr als tausend Worte!“ gilt global. Bilder sind international verständlich und sprachfrei, müssen also nicht in alle Sprachen der Welt übersetzt werden. Insbesondere technische Zusammenhänge, wie bei CAD-

Systemen überwiegend der Fall, sind durch grafische Darstellung kompakt, unmittelbar sowie allgemein und schnell verständlich zu dokumentieren. Die Qualität und Aussagekraft einer grafischen Darstellung ergibt sich allerdings nicht von allein durch die Natur der Sache, also der Grafik, sondern erst durch die gestalterischen Ideen und Überlegungen des Autors der Bilder.

Statische Grafik stellt einen konstanten Sachverhalt dar, die bildliche Darstellung bleibt unverändert. Aktive Grafik bietet darüber hinaus innerhalb der statischen Grafik die Möglichkeit, Dateneingaben direkt in der Grafik vorzunehmen, z. B. durch Eingabefelder anstelle von Maßzahlen einer Bemaßung, siehe Bild 4-15. Dies ist wesentlich aussagekräftiger, eindeutiger und zugeordneter als die in bisherigen CAD-Systemen üblichen Dialogfelder mit einer Kombination aus statischer Grafik und daneben angeordneter Liste von benannten Eingabefeldern. Die mangelhafte Zuordnung der Benennung eines Eingabefelds zur zugehörigen Funktion in der Grafik lässt oft Spielraum für Fehlinterpretation, abgesehen vom Übersetzungs-aufwand.

Im Sinne dieser Arbeit sind daher im CAD-System BOCAD-3D z. B. für die im Stahlbau oder Holzbau üblichen Anschlussarten Fenster mit aktiver Grafik konzipiert worden, sogenannte Leitbilder (Bild 4-15). Dabei wurde systematisch angestrebt, durch die Bildgestaltung und -anordnung für alle Leitbilder eine hohe Wiedererkennungchance für gleiche Sachverhalte zu erzielen (gleiches „Look and feel“) und mit wenigen übersichtlichen Darstellungen möglichst ohne Texte auszukommen. Der dafür aufzuwendende Ideenreichtum, aber auch die Gestaltungsdisziplin aller Beteiligten, war erheblich. Viele Diplomarbeiten [10 u. a.], die in direkter Zusammenarbeit mit führenden CAD-Ingenieuren der Baupraxis durchgeführt wurden, ergaben im Laufe mehrerer Jahre einen Fundus an Leitbildern, die den o. g. Kriterien nahe kommen.

An einem ausgewählten Beispiel sein die Funktionsweise und der Nutzen derartiger Leitbilder veranschaulicht.

In den Leitbildern von BOCAD-3D ist also Information, Dokumentation und Dateneingabe mit Hilfe von statischer, gestaltkonstanter Grafik und Eingabefeldern in der Grafik zusammengefasst. Bei statischer Grafik kann der Entwickler des

Leitbilds die Form der Objekte durch Übertreibungen so gestalten, dass die Lesbarkeit der Zeichnung bis ins Detail selbst bei naturgemäß gedrängter Anordnung optimal wird. Gerade im Stahlbau, Holzbau und Glasbau können Details sehr kleingliedrig und somit wegen Überfrachtung schlecht lesbar sein. Die Kombination statischer, nicht naturgetreuer Grafik mit aktiver Grafik, also in die Grafik am aussagekräftigsten Ort integrierten Eingabefeldern, bietet die umfassendsten und für den Anwender einfachsten Eingabemöglichkeiten. Dieser wesentliche Vorteil muss gewahrt bleiben, kann aber durch zusätzliche dynamische Grafik ergänzt werden.

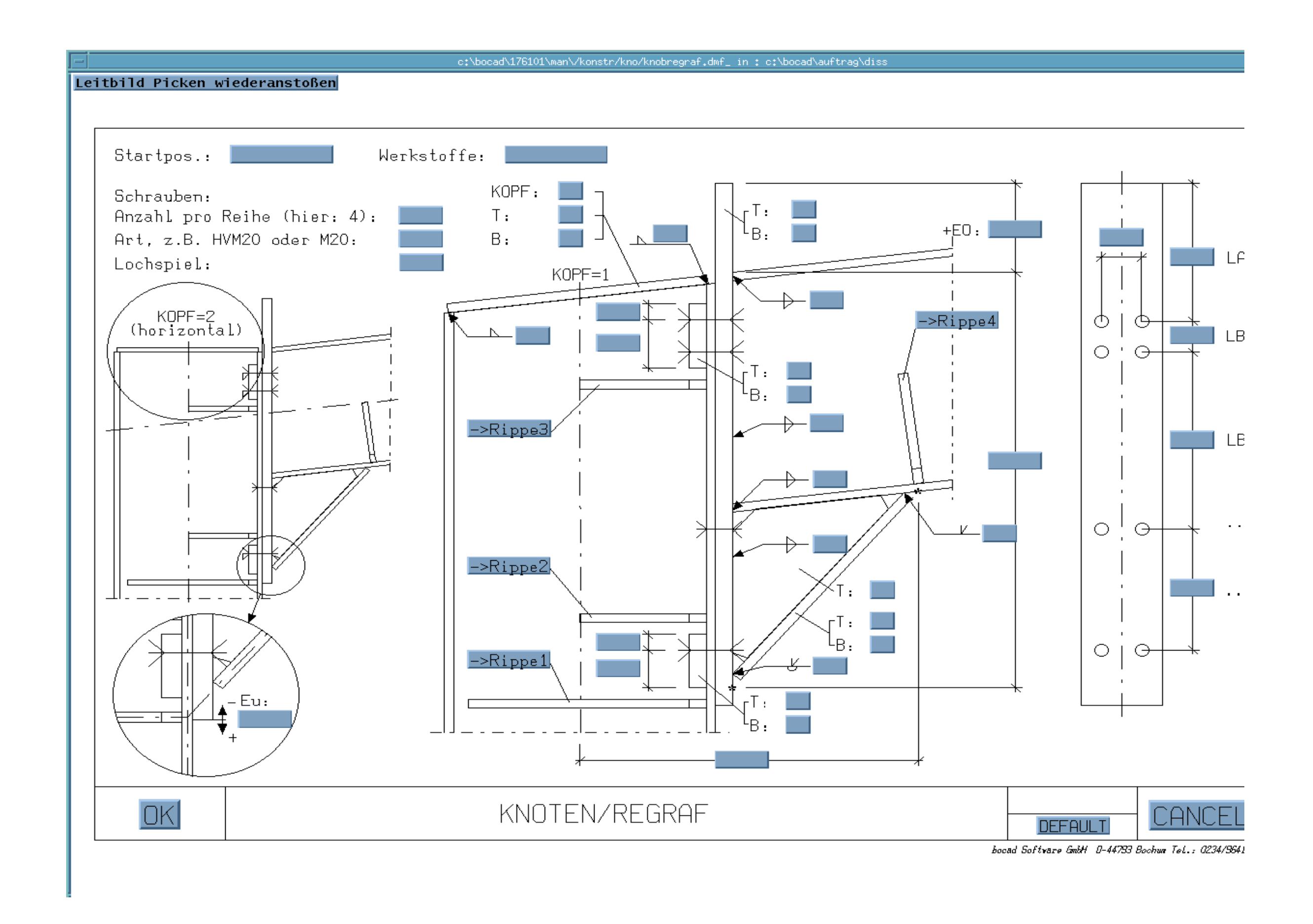
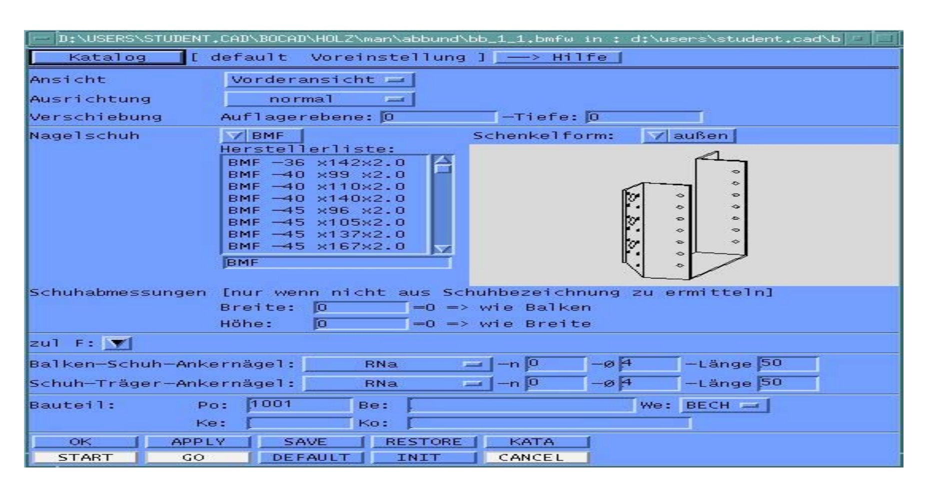


Bild 4-15: Leitbild „Rahmenecke Bauart Grafbau“, Werkbild BOCAD

Dynamische Grafik unterscheidet sich von statischer Grafik dadurch, dass sie in Abhängigkeit von jedem eingegebenen Wert sofort die entsprechende Wirkung durch dynamische Veränderung der Grafik anzeigt. Auch dynamische Grafik könnte bei erheblicher Steigerung des programmtechnischen Entwicklungsaufwands mit

aktiver Grafik ausgestattet werden. Die höchste Leistungsstufe im Sinne einer einfachen, anschaulichen Bedienung ist jedoch die Kombination von statischer Grafik mit bester Lesbarkeit und Eingabefeldern in dieser Grafik sowie zusätzlicher dynamischer Grafik mit naturgetreuer Gestalt. Ein Beispiel mit zwei Bauarten von Balkenschuhen, die als Grafik je nach Wahl des CAD-Ingenieurs angezeigt werden, veranschaulicht „dynamische Grafik“.



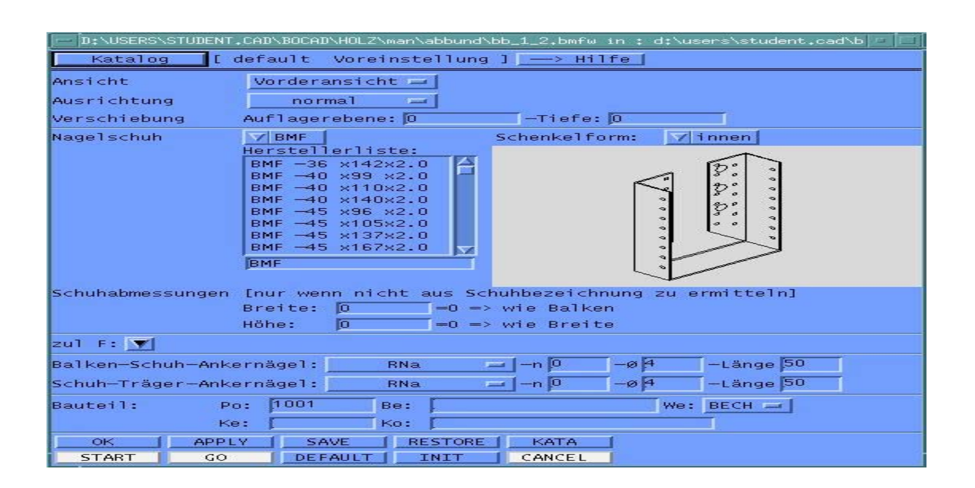


Bild 4-16: Dialogfeld mit dynamischer Grafik
(mit Balkenschuh-Schenkelform, „außen“ und „innen“, BOCAD-Holz)

Nicht ganz so hohe Entwicklungsanforderungen stellen Leitbilder, die mit einer Kombination aus dynamischer Grafik und daneben angeordneter Liste von benannten Eingabefeldern gestaltet sind. Sie sind zwar in vielen CAD-Systemen üblich, die Zuordnung eines Maßes zu einem benannten Eingabefeld ist dabei allerdings nicht immer offensichtlich, abgesehen von der Sprachgebundenheit.

Die Information über Objekteigenschaften und das Ändern von Objekteigenschaften sind auch in elementaren Zeichensystemen relativ häufig genutzte Aktionen. Beide Möglichkeiten werden daher zweckmäßig zusammengefasst. Führende Zeichensysteme sehen allerdings lediglich eine horizontale, schmale Eigenschaftsleiste mit objektspezifischen Elementen vor. Die Grafiken der Eigenschaftsleisten sind nur kleine Symbole, z. B. ° für Winkelangaben in Grad oder Rechtecksymbole für Querformat bzw. Hochformat.

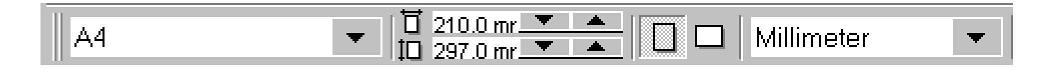


Bild 4-17: Eigenschaftsleiste für Papierformat in Corel Draw

Bei direkten Aktionen am Objekt wird z. B. die Länge des Spurvektors (Abschnitt 4.2.4) in der Eigenschaftsleiste dieser Zeichensysteme nicht mitlaufend angezeigt.

Auf Basis dieser Erfahrungen sollen für die in dieser Arbeit konzipierten Eigenschaftsfenster folgende Grundregeln gelten:

Generell werden die Eigenschaftsfenster so gestaltet, dass sie jede direkte Änderung am Objekt mitlaufend grafisch darstellen. Soweit irgend möglich, werden Erläuterungstexte oder Parameternamen vermieden, um durch selbsterklärende grafische Darstellungen das Sprachfreiheitsprinzip zu erfüllen. Letztlich werden dadurch Eigenschaftsfenster zu Leitbildern mit bidirektional mitlaufender Aktualisierung.

Eigenschaftsfenster sind naturgemäß objektabhängig. Die Beschreibung der konkreten Ausprägung einzelner Eigenschaftsfenster folgt deshalb in den Unterabschnitten des Kapitels 5, in denen die einzelnen Objekte und ihre Methoden entwickelt werden.

Alle bisher als geeignet erkannten Gestaltungselemente mit hohem Bekanntheitsgrad werden dabei verwendet, wie Eingabefelder, Kontrollfelder, Optionsfelder, Drehfelder, mitlaufende Wertanzeige sowie passive, aktive und dynamische Grafik.

4.2.9 Kontextmenüs

Üblicherweise erscheint in Softwareprogrammen nach einem Rechtsklick auf ein Objekt das objektspezifische Kontextmenü mit einer textförmigen Liste von Befehlen und Eigenschaftswerten. Das Kontextmenü zeigt also die Methoden des Objekts im Sinne der objektorientierten Programmierung.

Ziel dieser Arbeit ist es, textförmige Kontextmenüs möglichst vollständig durch symbolische, sprechende Funktionsgriffe direkt am Objekt zu ersetzen und Objekteigenschaften durch objektspezifische Eigenschaftsfenster anschaulich und änderbar zu machen.

4.3 Exemplarisches Beispiel programmtechnischer Realisierung

Nach den Analysen und dem Vergleich praxisüblicher CAD-Systeme wurde exemplarisch eine Nutzoberfläche für 2D-Bearbeitung entwickelt und in das zu Testzwecken zur Verfügung gestellte CAD-System BOCAD-3D integriert. In BOCAD-3D wurden mehrere hundert Mannjahre Entwicklung investiert. Es erfordert ein Ingenieurteam von 20 Personen zur laufenden Wartung und Weiterentwicklung. Deswegen wäre es ein extremes unternehmerisches Wagnis, das bestehende System mit den neuesten Werkzeugen der Informatik vollständig neu zu entwickeln. Es wird daher untersucht, wie eine Bausoftware bei laufendem Alltagseinsatz für den Nutzer wirtschaftlich und ohne negative Folgen durch „Re-Engineering“ ergänzt und weiter entwickelt werden kann. Dadurch wurden unvorhergesehene, große Technologiesprünge der Informatik und des Anwendungsumfelds abgefangen.

BOCAD-3D ist ein dreidimensional orientiertes CAD-Programm. Es leitet ein 3D-Modell aus deren Daten der Datenbank ab. Obwohl Entwurfszeichnungen direkt während der Erzeugung des 3D-Modells über ein Symbol (*Bild Unter Namen sichern*) gespeichert oder auch nach der Positionierung automatisch „just in time“ erzeugt werden können, ist die nachträgliche Bearbeitung einer 2D-Darstellung in manchen Fällen, z. B. bei Zukaufteilen aus 2D-Katalogen, unvermeidbar, siehe Kapitel 5.2.6. Um diese Leistungslücke abzudecken, wurden Grafikwerkzeuge für die Zeichnungsbearbeitung entwickelt.

Diese neue Oberfläche erfordert drei Leistungsgruppen - Grafik, Punkte und Bemaßen - für 2D-Grafik-Bearbeitung. Diese stehen über selbsterklärende Symbole direkt zur Verfügung, siehe Bild 4-18.

<table>
<tr><td rowspan="2">Grafik generieren</td><td colspan="3">Allgemeines (Datei, Öffnen, Sichern, Drucken… Darstellungsbezogene Funktionen...)</td><td rowspan="2">Punkte erzeugen,
Information,
Löschen</td></tr>
<tr><td colspan="3" rowspan="2">Bildbereich</td></tr>
<tr><td rowspan="2">Grafik ändern,
Löschen</td><td rowspan="2">Bemaßen,
Ändern,
Löschen</td></tr>
<tr><td>Textattribute</td><td>Linienattribute</td><td>weitere Eigenschaften</td></tr>
</table>

Bild 4-18: Bildbearbeitung und Symbolanordnung für 2D-Grafik

4.3.1 Wählen von Symbolen - Vermeidung von Eingaben und von „Mehr-Klassen-Menüs“

Im zur Verfügung gestellten CAD-System kann über einen Klick auf die rechte Maustaste das zu einer Funktion oder Leistung gehörende Dialogfenster aufgerufen werden. Durch Wählen von Optionen bzw. durch Eingabe von Werten kann der Anwender das Ergebnis der folgenden Konstruktion steuern. Das ist z. B. notwendig beim Generieren eines 3D-Objekts, z. B. einem Profil: Der Anwender muss zunächst das Profil aus einer Liste auswählen, die Transformationswerte für Quer- und Tiefenverschiebung und die Rotation festlegen, um das Teil platzieren zu können.

Für 2D-Grafikobjekte sind Angaben zur Platzierung im Raum irrelevant. Anhand eines geschlossenen Polygons soll dies veranschaulicht werden. Bevor der erste Punkt des Polygons gewählt werden kann, müssen folgende Schritte abgearbeitet werden:

1. Menü *Grafik* über die linke Maustaste wählen;
2. Eintrag *Polygon* klicken;
3. Mit der Rechtstaste der Maus das zugehörige Dialogfenster öffnen;
4. Ausführungsart des Polygonzugs *geschlossen* auswählen;
5. Schaltfläche *OK* wählen.

In der neuen, direkten Oberfläche braucht dagegen nur einmal das Symbol  *geschlossenes Polygon* gewählt zu werden.

Außerdem problematisch war die Tatsache, dass im bereitgestellten System manche Leistungen in einem verschachtelten „Mehr-Klassen-Menü" versteckt waren. Um eine solche Leistung aufzurufen, musste der Anwender mehrere Male klicken, bevor er sie erreichte. Zur Ermittlung des Abstandes zweier Punkte beispielsweise mussten drei Schritte ausgeführt werden:

1. Menü *Punkte* wählen;
2. Eintrag **Testmaß** wählen;
3. Im Unterklassenmenü den Eintrag „*Abstand von 2 Punkten*" wählen. Dann erst konnte der Benutzer die zwei Punkte wählen.

Im Vergleich dazu kann der Benutzer nun aufgrund der hier vorgestellten Entwicklung den erheblich einfacheren Weg gehen, in dem er das Symbol verwendet.

Die Leistungen für „Grafik Generieren" in der neuen Oberfläche wurden neu eingeteilt. In der alten Oberfläche mussten Parameter im Dialogfenster festgelegt werden. Tabelle 4-4 zeigt die neue direktere Einteilung der Leistung für „Grafik Generieren".

Altes Menü	*Generierungsart*	*Sprachfreie Symbole*
Linie:	2-Punkte-Form	
	Punkt-Winkel-Form	
	dx-dy-Form	
	Linie	
Linie/n parallel:	***Parallele Kopie des Originals über***	
	Pickpunkt	
	Querverschiebungswert	
Polygon:	offen	
	geschlossen	
	Gefüllt	
Regelmässiges Polygon:	***Zeichenmodus***	
	Zur Ecke	
	Einzelne Seite	
	Zur Seite	
Rechteck:	Diagonale	
	Einzelne Seite und Höhe über Pickpunkt	
	Einzelne Seite und Höhe über Wert	
	Höhe und Breite	
Kreis:	Kreis über Radius	
	Kreis über Punkt	
	Kreisbogen	
Linie -> Verlängern Verkürzen:	**Ändernmodus**	
	Über Abstand	
	Über Pickpunkt	
Linienpunkte ziehen:	Pickelement — Verschiebung über	
	Linie — Punkt	
	Linie — Wert	
	Punkt — Punkt	
	Punkt — Wert	

Tabelle 4-4: Sprachfreie Nutzoberfläche

4.3.2 Vereinfachung durch Voreinstellung

Bei jedem Aufruf einer Leistung aus dem Menü „Grafik“ oder „Bemaßen“ musste bisher der Anwender im zugehörigen Dialogfenster überprüfen, ob Linien- und Textattribute sowie andere allgemeine Attribute seine Wünsche erfüllen.

In der neu entwickelten Oberfläche wird der Inhalt der Dialogbox vereinfacht. Text- und Linienattribute sowie andere Eigenschaften, die zu diesem Zeitpunkt erforderlich sind, stehen nach den in Kapital 4.2 entwickelten Grundsätzen als Eigenschaftsleiste direkt unter dem Zeichnungsfenster, damit der Anwender die allgemeinen Attribute ständig im Blickfeld hat und somit leicht kontrollieren kann. Offensichtlich ist das Dialogfenster des Bildes 4-19 nicht sprach- und textfrei, wie im Thema der Arbeit gefordert. Die notwendigen Entwurfs- und Programmierkenntnisse für ein entsprechendes Re-Engineering waren jedoch nicht innerhalb der ersten drei Jahre der Promotionszeit zu erwarten.

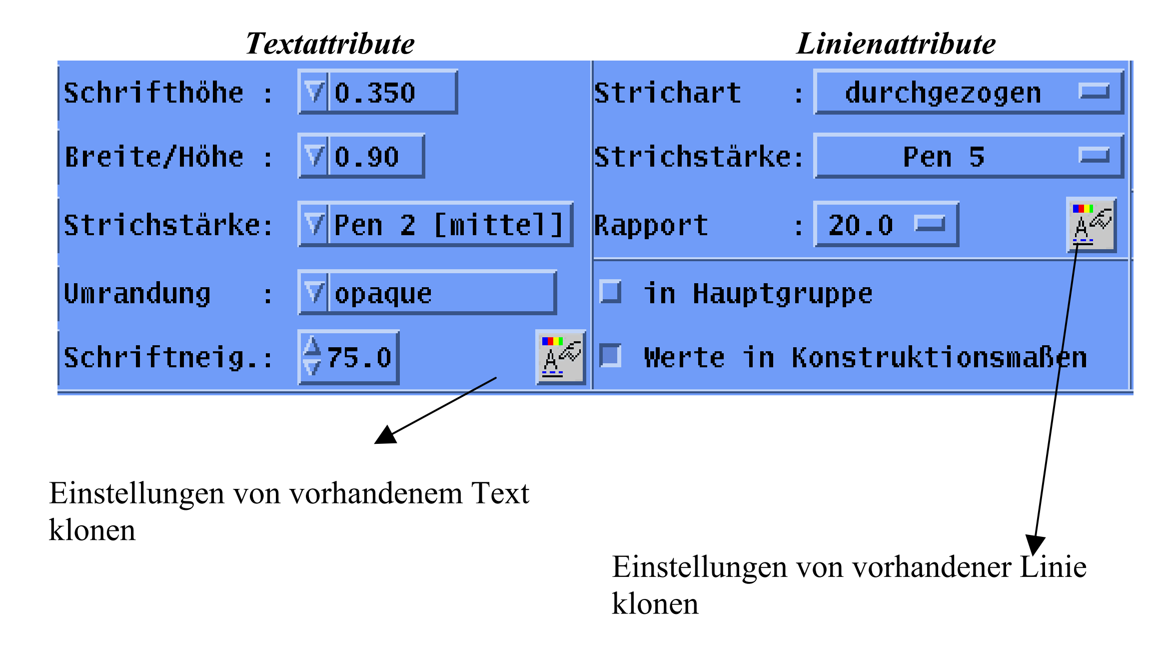
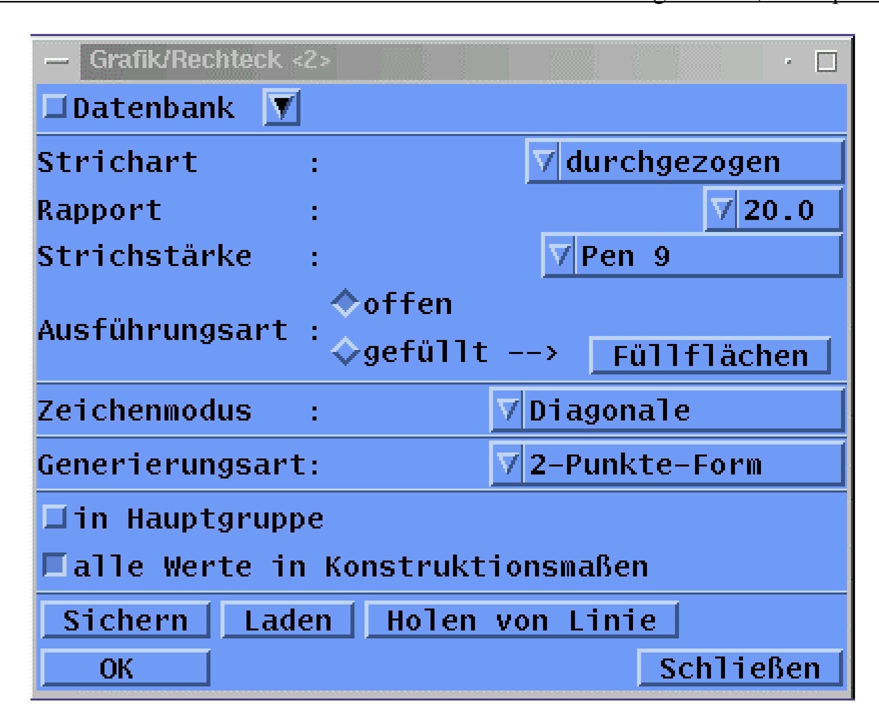


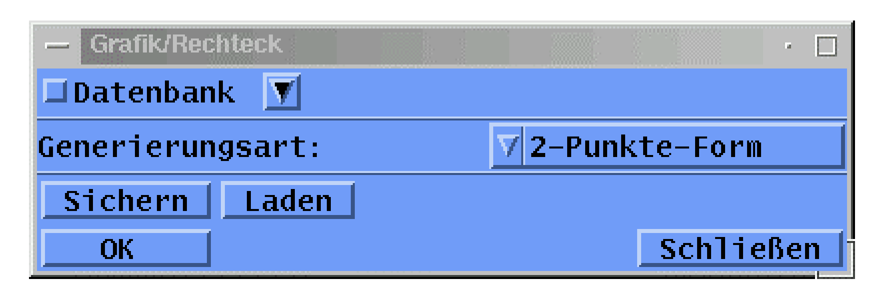
Bild 4-19: Allgemeine Grafikattribute

Zwei Veränderungen der Dialogfelder in dieser neuen Oberfläche wurden vorgenommen:

- Dialogfenster werden deutlich vereinfacht, in manchen Fällen kann der Anwender sogar direkt Grafikelemente zeichnen, ohne eine Dialogbox mit der rechten Maustaste aufzurufen, siehe Bild 4-20.



Bisheriges Dialogfenster für das Zeichnen eines Rechtecks



Neuentwurf des Dialogfensters für Rechteck-Zeichnen

Bild 4-20: Vergleich alter und neuer Dialogfenster für Rechteck-Zeichnen

- Das mehrfach gestufte Mehrklassensystem der Dialogfenster wurde aufgehoben. Vorher haben manche Dialogboxen bis zu zwei Unterklassen enthalten. Dieser komplizierte Aufbau wird in der neuen Oberfläche vermieden. Ein Beispiel zeigt die vereinfachten Dialogfenster für die Steigungsdreieck-Bemaßung (siehe Bild 4-21 und 4-22).

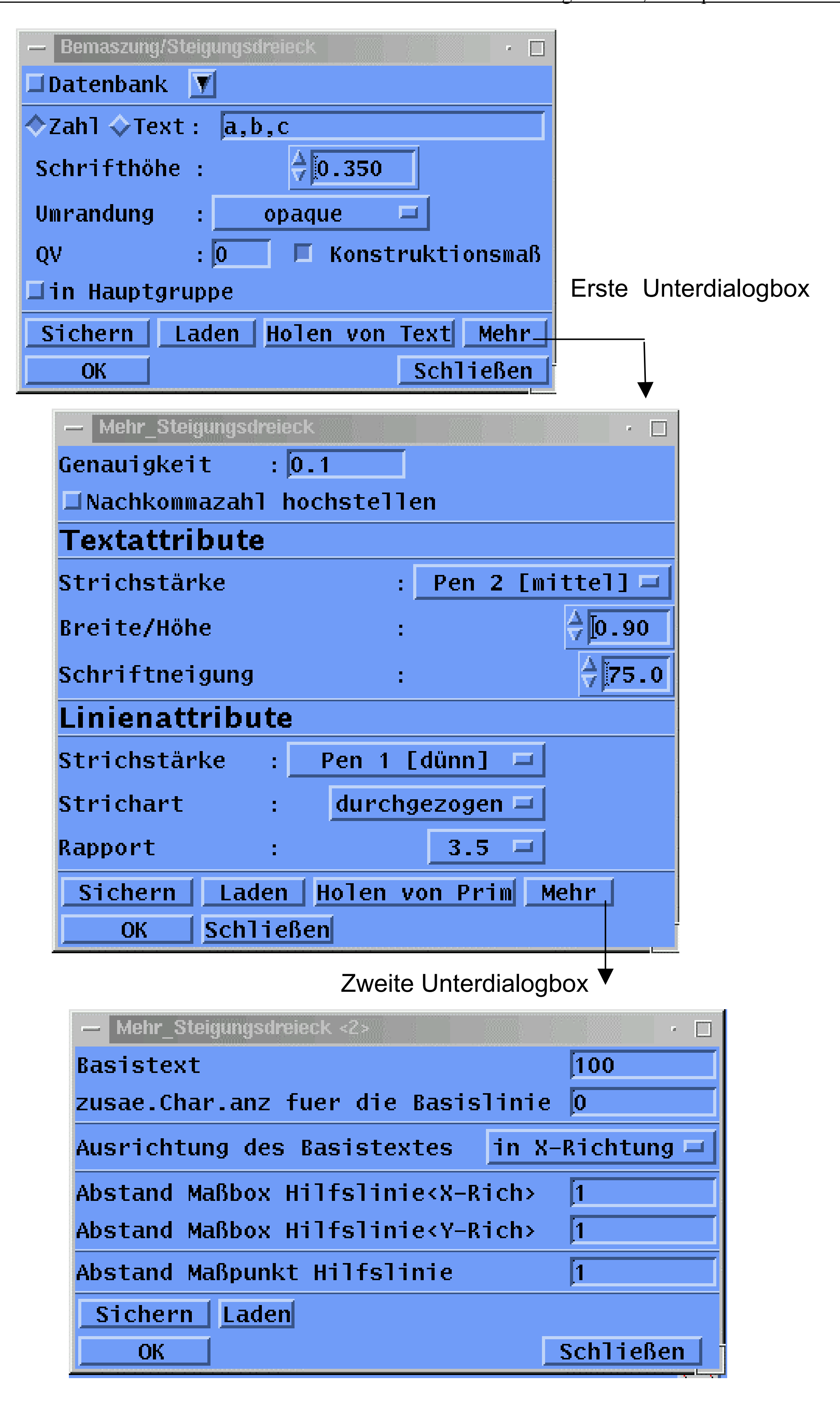
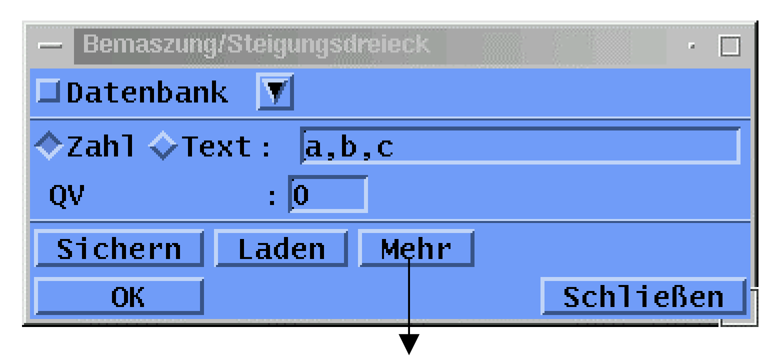


Bild 4.21: Bisherige Dialogfenster für die Steigungsdreieck-Bemaßung



Nur eine Unterdialogbox

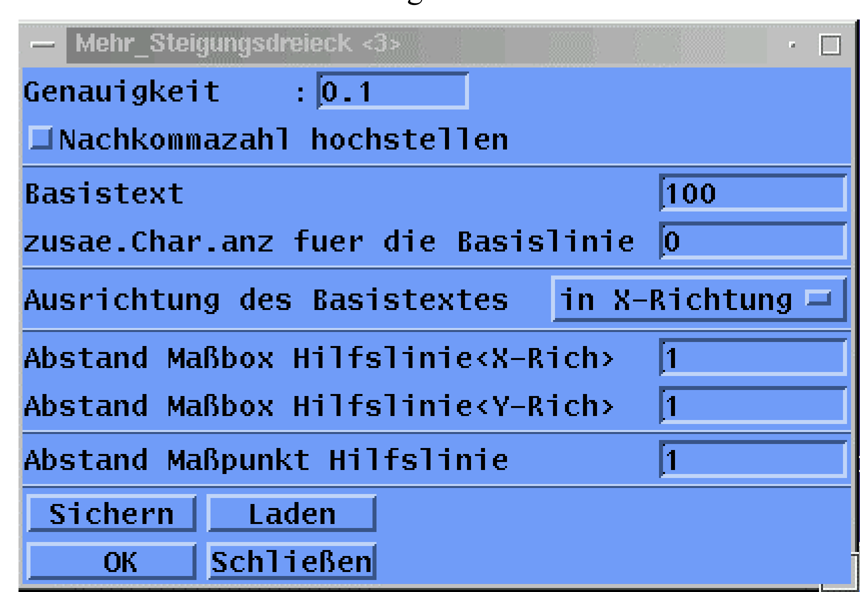


Bild 4-22: Neue Dialogfesnter für Steigungsdreieck-Bemaßung

4.3.3 Integration in ein bestehendes System durch Re-Engineering

Die neue Oberfläche für 2D-Grafik-Bearbeitung bzw. für den Papierbereich wurde, die Gewohnheiten des Benutzers erhaltend, gegenüber der bisherigen Oberfläche kaum verändert. So bleiben „alte“ Dialogfenster erhalten, aber über die „neue“ 2D-Grafikbearbeitungsumgebung werden „neue“ Dialogfenster aufgerufen, obwohl identische Leistungen durchgeführt werden.

Durch ein neues Programmmodul „**parsereq**“ wurde es ermöglicht, verschiedene Menüs oder Symbole aufzurufen, welche die gleiche Funktion bzw. den gleichen Callback durchführen. Diese jedoch können dann verschiedene Dialogfenster bzw. *RechteMausCallbacks* aufrufen. Darüber hinaus werden Werte bzw. allgemeine Attribute, die im unteren Fenster als Eigenschaftsleiste mit neuen Adressen stehen,

vor der Durchführung der unveränderten Kernfunktion in die alten Adressen eingesetzt.

Im Folgenden wird der neu entwickelte Standardcallback BoInstallReqGrp <NrRequest, TypRequest> für die Linien-Generierung vorgestellt:

grafik.rqi

```
/* ##########
/* GENERIEREN: Linie
/* ##########
/* -->> Grafik: 2-Pkt-Form
Begin:65006610,23510000;
 BeginCallback:1,BoPrePowerGrf;
 RechteMausCallback:1,BoCallWasnach;
 ShiftRechteMausCallback:1,BoCreateCombiBox<1500:12351:1505>;
 CtrlRechteMausCallback:1,BoCreateCombiBox<1500:12351:1509>;
 BeginReq;
  RequesType:101;
  CursorType:102;
  C2:2103;
  MinElReq:2;
  MaxElReq:2;
  MessFileNr:3500;
  MessNr:-1;
  Messages:1,2,2;
  PickMode:25604;
  Securitylevel:-2;
  ErgUsnr:1;
  ErgUstp:18069900;
  PrevUsnr:66100000;
  PrevUstp:20899000;
  CoorSource:10;
  CompareVal:1;
  CompareAdr:1 12351 graf bas2 default 40 4;
  Callback:1,GenLine<1:1:-1>;
 EndReq;
/* -->> Grafik: dx-dy-Form
 BeginReq;
  RequesType:101;
  CursorType:102;
  MinElReq:1;
  MessFileNr:3500;
  MessNr:30;
  PickMode:25604;
  Securitylevel:-2;
  ErgUsnr:1;
  ErgUstp:18069900;
  PrevUsnr:66100001;
  PrevUstp:20899000;
  CompareVal:2;
  CompareAdr:1 12351 graf bas2 default 40 4;
  Callback:1,GenLine<2:1:-1>;
 EndReq;
/* -->> Grafik: Punkt-Winkel-Form
 BeginReq;
```

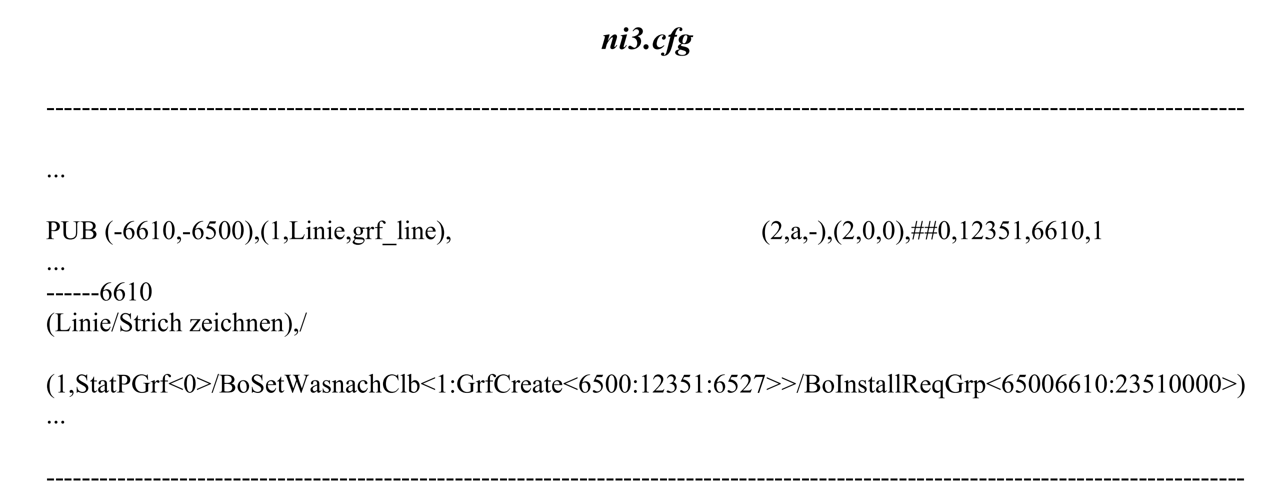
```
  RequesType:101;
  CursorType:102;
  MinElReq:1;
  MessFileNr:3500;
  MessNr:30;
  PickMode:25604;
  Securitylevel:-2;
  ErgUsnr:1;
  ErgUstp:18069900;
  PrevUsnr:66100002;
  PrevUstp:20899000;
  CompareVal:4;
  CompareAdr:1 12351 graf bas2 default 40 4;
  Callback:1,GenLine<3:1:-1>;
 EndReq;
/*
/* -->> Grafik: Linien-Form
 BeginReq;
  RequesType:111;
  CursorType:102;
  MinElReq:1;
  MaxElReq:1;
  MessFileNr:6500;
  MessNr:4;
  PickMode:51204;
  Securitylevel:1;
  ErgUsnr:1;
  ErgUstp:18069900;
  PrevUsnr:66100003;
  PrevUstp:20899000;
  CompareVal:8;
  CompareAdr:1 12351 graf bas2 default 40 4;
 EndReq;
 BeginReq;
  RequesType:101;
  CursorType:102;
  MinElReq:1;
  MessFileNr:6500;
  MessNr:71;
  PickMode:1;
  Securitylevel:-2;
  ErgUsnr:2;
  ErgUstp:18069900;
  PrevUsnr:66100004;
  PrevUstp:20899000;
  CompareVal:8;
  CompareAdr:1 12351 graf bas2 default 40 4;
  Callback:1,GenLine<4:1:2>;
  PCallback:1,ClbFileSelectInd<1:0>;
 EndReq;
End;
/*
```

Im „BeginCallback“ wird die Funktion BoPrePowerGrf durchgeführt. Mit dieser wird überprüft, ob diese Funktion in der neuen Oberfläche aufgerufen wird. Sollte dies der Fall sein, werden die neuen Adressen für die allgemeinen Attribute in die alten

Adressen eingesetzt.

Darüber hinaus fällt beim RechteMausCallback auf, dass statt eines gewöhnlichen Dialogfensters hier eine Funktion BoCallWasnach eingesetzt wird. Diese Funktion ermöglicht bei Aufruf einer identischen Leistung, verschiedene Dialogboxen durch BoSetWasnachClb öffnen zu lassen.

In der Datei *ni3.cfg* werden drei Callbacks beim Starten der Funktion **Grafik --> Linie** aufgerufen:



ni3.cfg

```
--------------------------------------------------------------------------------

...

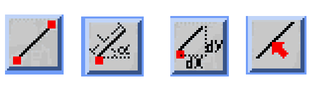
PUB (-6610,-6500),(1,Linie,grf_line),                    (2,a,-),(2,0,0),##0,12351,6610,1
...
------6610
(Linie/Strich zeichnen),/

(1,StatPGrf<0>/BoSetWasnachClb<1:GrfCreate<6500:12351:6527>>/BoInstallReqGrp<65006610:23510000>)
...

--------------------------------------------------------------------------------
```

Im Callback StatPGrf<0> wird die neue Oberfläche ausgeschaltet. Durch BoSetWasnachClb<1:GrfCreate<6500:12351:6527>> wird GrfCreate <6500: 12351: 6527> - die alte Dialogbox für **Linie-Zeichnen** - als der RechteMausCallback eingestellt; Der Callback BoInstallReqGrp<65006610:23510000> ruft einen Standardcallback auf, der schon im vorherigen in der Datei *grafik.rqi* erläutert worden ist.

Auch für die Funktionen der Linien-Erzeugung, die über die nachstehenden Symbole zu erreichen sind, wurden in der Datei *power_grf.cfg* drei Callbacks verwendet:



power_grf.cfg

```
--------------------------------------------------------------------------------

...

PUB (1103,1710,1, 3),(0,linie_2pkt.xpm,grf_line),   (2,a,-),(Linie: 2-Punkte-Form), (2,1,0), /
                    (1,StatPGrf<1|6610|1>/BoSetWasnachClb<1:GrfCreate<6503:12351:65270>>/
                                   /BoInstallReqGrp<65006610:23510000>)
```

```
PUB (1104,1710,1, 4),(0,linie_lenwin.xpm,grf_line),(2,a,-),(Linie: Punkt-Winkel-Form),(2,1,0),/
                    (1,StatPGrf<1|6610|2>/BoSetWasnachClb<1:GrfCreate<6503:12351:65272>>/
                                  /BoInstallReqGrp<65006610:23510000>)
PUB (1105,1710,1, 5),(0,linie_dxdy.xpm,grf_line),    (2,a,-),(Linie: dx-dy-Form),(2,1,0),/
                    (1,StatPGrf<1|6610|4>/BoSetWasnachClb<1:GrfCreate<6503:12351:65271>>/
                                  /BoInstallReqGrp<65006610:23510000>)
PUB (1106,1710,1, 6),(0,linie_klick.xpm,grf_line),   (2,a,-),(Linie: Linie picken-Form),(2,1,0),/
                    (1,StatPGrf<1|6610|8>/BoSetWasnachClb<1:GrfCreate<6503:12351:65273>>/
                                  /BoInstallReqGrp<65006610:23510000>)
...

------------------------------------------------------------------------------------------------------------------------------------------
```

Über den Callback StatPGrf<1|6610|1> wird dann die neue Oberfläche eingeschaltet. Dann wird der Index für die aufgerufene Funktion 6610 - Linie Generieren - eingesetzt. Schließlich wird der Index für die Einteilung der Leistung gemäß der Generierungsart oder dem Zeichenmodus eingestellt. Durch BoSetWasnachClb <1:GrfCreate<6503:12351:65270>> wird GrfCreate <6503:12351:65270>> - das neue Dialogfenster für **Linie-Zeichnen** - als RechteMausCallback eingestellt.

4.4 Programmtechnische Konzeption unter der Randbedingung des Re-engineering größerer CAD-Systeme im Alltagseinsatz

Re-Engineering großer CAD-Systeme im Alltagseinsatz gibt erschwerende Entwicklungsrandbedingungen vor.

Revolutionäre Entwicklungssprünge, die eine komplette Neuprogrammierung eines gesamten CAD-Systems in einem Zuge bedingen, verbieten sich von selbst. Die Umstellung von Algorithmen, die mit einem Aufwand von mehr als hundert Mannjahren entwickelt wurden und ausreiften, sind in Entwicklungsunternehmen in aller Regel nur in überschaubaren, d. h. in finanzierbaren Schritten zu verwirklichen.

Zum Re-Engineering kommen also nur Technologien in Betracht, die eine schrittweise Erneuerung im laufenden Betrieb ermöglichen.

Um nicht in völlige Abhängigkeit des Betriebssystems Windows zu geraten, soll die programmtechnische Konzeption so gestaltet werden, dass ebenso wie Windows auch das technisch effizientere Betriebssystem LINUX verwendet werden kann. Die Randbedingung lautet daher kurz: Unabhängigkeit von Plattformen. Für die Erfüllung dieser Randbedingung wird als Entwicklungswerkzeug die Software QT eingesetzt, die Elemente der gewohnten Windows Nutzoberfläche in Form von Oberflächenobjekten in der Sprache C++ sowohl unter Windows als auch unter LINUX zur Verfügung stellt.

Das hier untersuchte CAD-System BOCAD ist über viele Jahre im Praxiseinsatz ausgereift und das System hat sich mit großem Markterfolg bewährt, bleibt aber nicht automatisch nachhaltig technisch führend. Es ist daher eine typisch ingenieurwissenschaftliche Aufgabenstellung, eine Lösung zur Erneuerung der ausgereiften Software-Systeme zu erforschen. Im Folgenden werden einige CAD-Objekte bzw. Verbesserungsmöglichkeiten, die in der Konstruktionspraxis gewünscht wurden, im Experiment ingenieurwissenschaftlich nachgewiesen.

Ein vollständiges Re-design mit jeweils neuesten Werkzeugen der Informatik ist nur eine sinnvolle Lösung bei kleineren Systemen. Bei großen Systemen mit entsprechendem Leistungsumfang, in die mehrere hundert Mannjahre Entwicklung investiert wurden und die große Ingenieurteams zur laufenden Wartung erfordern, ist

dieser Weg ein extremes unternehmerisches Wagnis. Es wird daher auch untersucht werden, wie große Bausoftware-Firmen bei laufendem Alltagseinsatz durch Re-engineering die Verbesserungsmöglichkeiten bzw. die im praktischen Umgang gebrauchten Leistungen realisieren.

Beim Re-engineering spielt der Stand der Entwicklungstechnik eine besonders wichtige Rolle. Ein effektives, qualitativ höherwertiges und stabileres System kann nur dadurch realisiert werden. Die Tabellen 4-5 und 4-6 zeigen die Verwaltung und den Prozess des Re-engineerings des untersuchten CAD-Systems BOCAD-3D mit führenden Entwicklungswerkzeugen:

Anforderungen verfolgen

Analysieren derzeitiger Technologiesprünge der Informatik → UML[1], Objektorientierte Sprache (Java, C++, …) und Datenbank

Bewerten der Märkte in den Stahlbau CAD Systemen

Kundenanforderungen verfolgen

↓

Werkzeuge analysieren und einsetzen

Modell: Together; Code: Together, Nedit, VC++ [2]

IDE: Eclipse, cForge,Kdevelop [3]; GUI: QtDesigener [4], Together, VC++

Debug-Programm: Together, ddd, VC++

↓

Koordination

Plattform Server: bocad37.bocad.de

↓

Dokumentation

Wergzeuge: Together, ctags, DoxyGen [5]

↓

Qualität zusichern

Statisch, dynamisch und semantisch kontrollieren; Abläufe sichern; Anwendung von Werkzeugen sichern

Über Änderungen informieren und Selbstkontrolle üben

↓

Projekt verfolgen

Produkte liefern, Schulung, Wartung

Tabelle 4-5: Verwaltung des Re-Engineering von BOCAD

----

[1] UML: Unified Modeling Language, die 1997 von der OMG (Objekt Management Group) als Standard akzeptiert wurde.

[2] Together, Nedit, VC++: 3 Entwicklungswerkzeuge, die zur Generierung des Programmscodes dienen können.

[3] IDE: Eclipse, cForge, Kdevelop: Integrated Development Environment und die 3 dazugehörigen Entwicklungsumgebungen.

[4] GUI: QtDesigener: GUI: graphical user interface; QtDesigener: Ein Wergzeug für GUI-Entwicklung

[5] ctags, DoxyGen: Wergzeuge für Dokumentation-Generieren

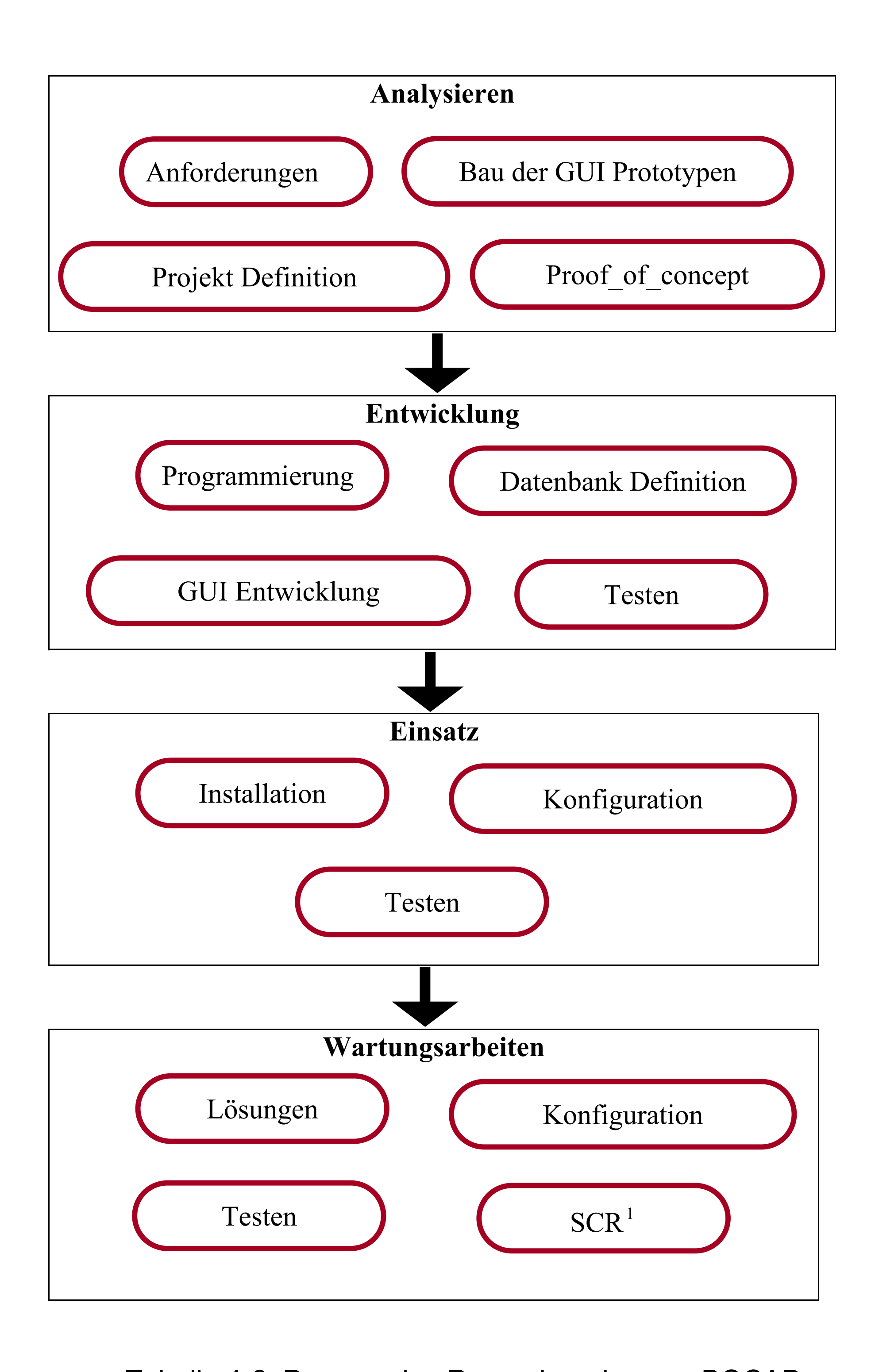


Tabelle 4-6: Prozess des Re-engineering von BOCAD

---

[1] SCR: Sequence Control Register

Die Oberfläche und die dahinter versteckte Konfigurations-Ressource (.cfg Format Datei) wird durch GUI (Graphical User Interface) – Entwicklung u. a. mit XML und Java erstellt. Zur Laufzeit des Programms wird die Konfigurations-Ressource interpretiert, siehe folgendes Bild 4-23.

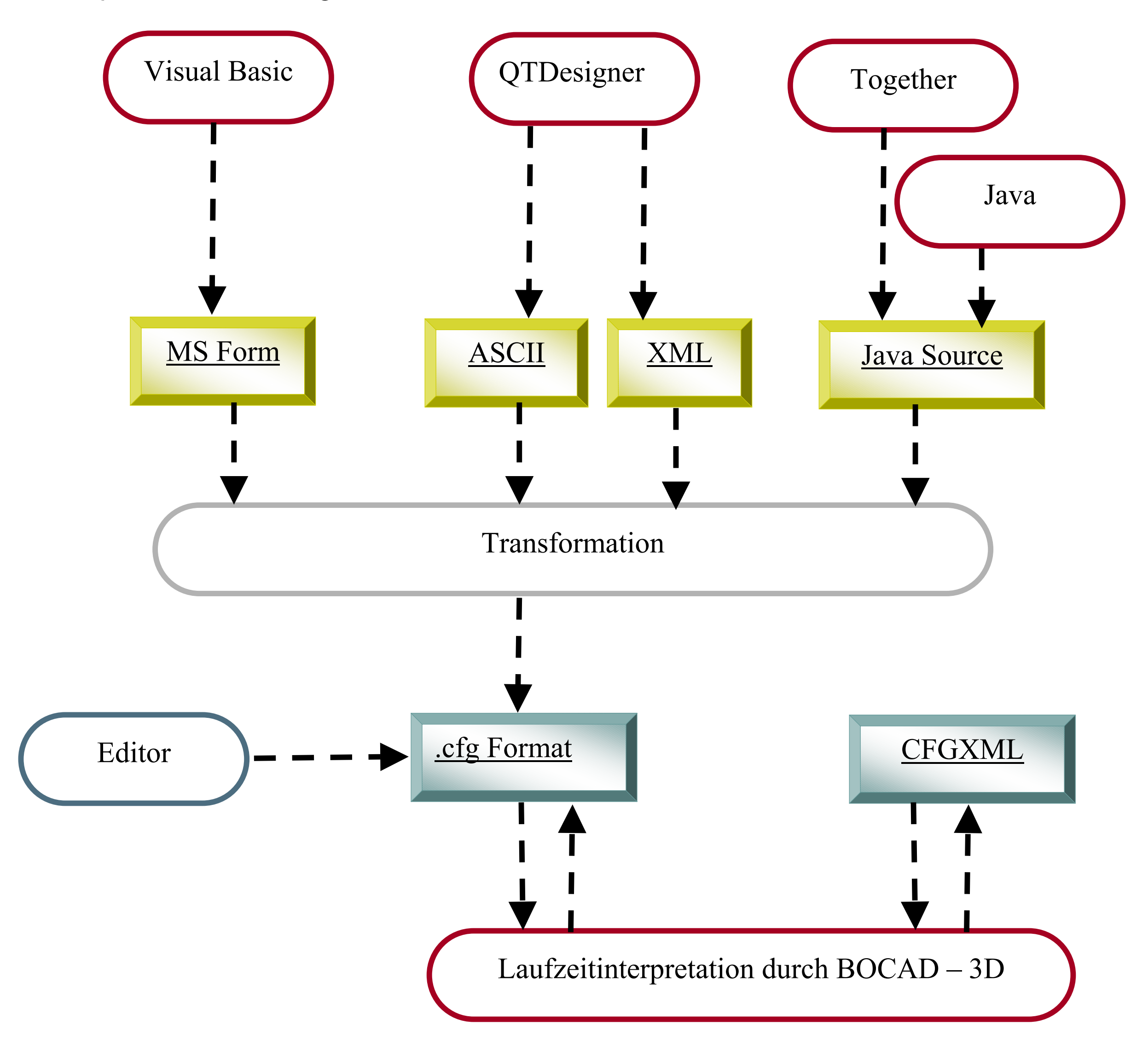


Bild 4-23: GUI Entwicklung

4.4.1 Re-engineering „intelligenter 2D-Objekte“: Kreis über Radiusliste und Kreis über 3 Punkte

Die folgenden drei Kreiskonstruktionsmöglichkeiten wurden von dem untersuchten CAD-System BOCAD angeboten:

- Kreis über Radius

  Ein Kreis wird durch den Mittelpunkt und den Radius festgelegt. Diese einfachste Form wird interaktiv durch Picken des Mittelpunktes und Angabe des Radius in einem Eingabefeld erstellt.

- Kreis über Punkt

  Ein Kreis wird durch den Mittelpunkt und einen Umfangspunkt des Kreises festgelegt. Es werden also interaktiv der Kreismittelpunkt und ein beliebiger Kreisumfangspunkt gepickt. Der vom Umfangspunkt bestimmte Radius wird gummibandartig angezeigt und nach Loslassen der Maustaste wird der Kreis gezeichnet.

- Kreisbogen

  Ein Kreisbogen wird durch den Mittelpunkt, einen Kreisbogenanfangspunkt und einen Kreisbogenendpunkt festgelegt.

Nach Erfahrungen der Baupraxis werden noch zwei weitere Konstruktionsmöglichkeiten gefordert:

- Kreis(bogen) über Radiusliste

  Mehrere konzentrische Kreise bzw. Kreisbögen werden durch Picken des Mittelpunktes und Eingabe einer Radiusliste in einem Eingabefeld festgelegt.

- Kreis(bogen) über 3 Punkte

  Ein Kreis wird durch drei Umfangspunkte des Kreises festgelegt. Es sind drei beliebige Kreisumfangspunkte wählbar, die jedoch nicht zusammenfallen dürfen.

Bei der Realisierung dieser zusätzlich gewünschten 2D-Leistungen wird zunächst die Oberfläche bzw. das Dialogfenster entworfen. Bild 4-24 zeigt das dazu konzipierte Dialogfenster für „Grafik/Kreis". In dem Listenfeld *Zeichenmodus* werden jetzt 6 Möglichkeiten der Generierung eines Kreises im Vergleich zu den alten 3 Generierungsarten angeboten.

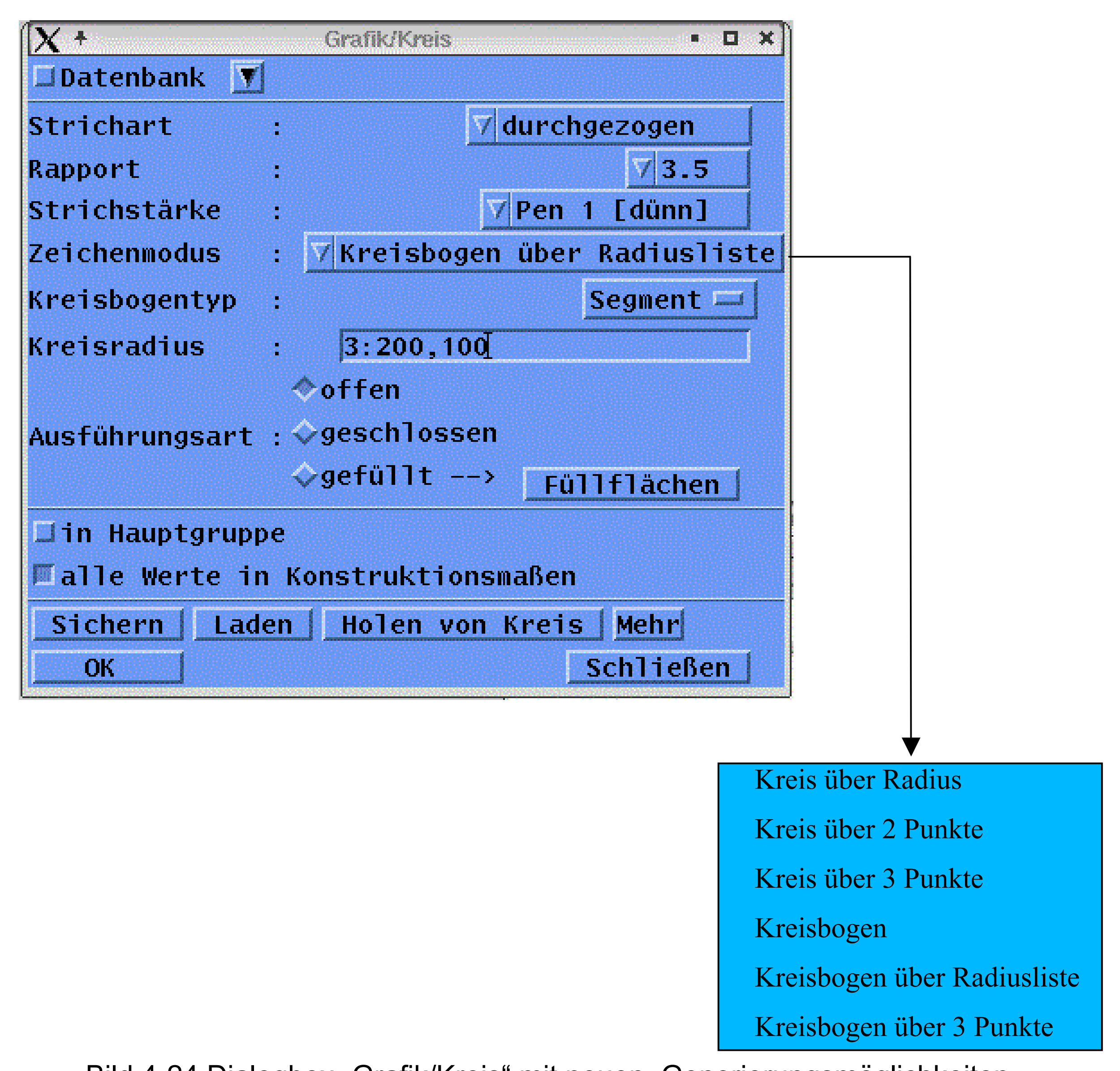


Bild 4-24 Dialogbox „Grafik/Kreis" mit neuen Generierungsmöglichkeiten

In dem Eingabefeld, das nach Auswahl „Kreis über Radius" erscheint, konnte der Benutzer bisher nur einen Einzelradius eingeben. Diese Leistung wurde so verbessert, dass eine Radiusliste eingegeben werden kann, d. h. mehrere konzentrische Kreise können mit einem Mal erzeugt werden. Die Oberfläche bleibt in

diesem Fall unverändert, aber dahinter versteckte Routinen müssen neu programmiert werden.
Die Entwicklungsschritte sind für diese Verbesserungsmöglichkeiten so durchzuführen:

· Einstellung der Eingabe für eine Radiusliste

Die Eingabemöglichkeit wird jetzt um eine Kombination mit Zahlen, Kommas und Doppelpunkten erweitert, beispielsweise 3:200, 100.

· Speichern und Abholen der Liste aus dem Eingabefeld

Die vom Anwender eingegebene Liste wird als *string* auf einer Adresse gespeichert. Diese Adresse wird in der Konfigurationsdatei *grafik.cfg* definiert. Diese Konfigurationsdatei **.cfg* dient dem Aufbau der Oberfläche von BOCAD.

-- grafik.cfg --

```
EDI[1] (3001,3000),(1,Kreisradius    :,grf_krad),\
   (1,12351,krei,2...,default ,16,20),\
   (1,12351,krei,2...,default ,16,20),\
   (3,34,10),        (0,1000000,1), (1,12351,zahl,1   ,default ,16,4)
```

---

[1] *EDI*: Ein Element von BoCreateCombobox, besteht aus:

(id),(name),(defadr),(tgtadr),(typawidth)

if <typ> == 1 (margins),(activatecallback),(Pushbuttoncallback)

if <typ> == 2 (intrange),(margins),(activatecallback),(Pushbuttoncallback)

if <typ> == 3 (dblerangeunit),(unitdefadr),(margins),(activatecallback),(Pushbuttoncallback)

if <typ> == 4 wie 1 aber führende Blanks nicht kappen

if <typ> == 5 (datumsteuerung) Datumsfeld

<typ>=(-2,-3) wird automatisch ein Pushbutton erzeugt, der den HP-Taschenrechner bringt.

<typ>=(-12,-13) wird automatisch ein Pushbutton erzeugt, der den TI-Taschenrechner bringt.

Dann wird diese Liste durch das Ausführungsprogramm *grafdo.c* von dieser Adresse abgeholt, wobei über case-Befehle der Sprache C die einzelnen Fallunterscheidungen getroffen werden.

```
---------------------------------------- grafdo.c ------------------------------------------------------
switch ( boline.k_art )
 { case  1: /* Kreis ueber Radiusliste */
          apst_such_keys ( " 1 12351 krei2...default  16 20", string , (char *) NULL);
          break;
   case  5: /* Kreisbogen ueber Radiusliste */
          apst_such_keys ( " 1 12351 krei3...default  16 72", string , (char *) NULL);
          break;
   default: /* weissnicht */
          break;
 }
--------------------------------------------------------------------------------------------------------
```

· Bearbeiten der Radiusliste

Ob die Liste syntaktisch und semantisch korrekt ist, wird danach durch das C Programm überprüft. Ist es der Fall, wird die Liste in einem Array *kreisradius* gespeichert. Siehe folgender Code.

```
---------------------------------------- grafdo.c ------------------------------------------------------
 if (boline.k_art == 1 || boline.k_art == 5)
    /* Kreis ueber Radiusliste oder Kreisbogen ueber Radiusliste*/
  {BoRemLTBlanks ( string );   /*Leer Zeichen weg*/
          lel = strlen(string);
          GetDimEck (&lel,string,&nradius, &status);  /*   Ermittlung der Anzahl des Radius-Arrays */
          if ( status )  goto ret;
          kreisradius    = (double *)malloc(  nradius  * sizeof(double) );
          if( (status=BoMallocErr(kreisradius,__FILE__,__LINE__)) && status) goto ret;
          GetEck ( string, &fac, kreisradius, &nradius, &eins, &status," ");/*   Ermittlung des Radius- Arrays */
```

```
        if ( status ) goto ret;
  }
```

· Zeichnen der Kreise

Die Kernfunktion für die Generierung eines Kreises ist *gzkre_ (float *x0, float *yy0, float *x1, float *yy1, float *x2, float *y2)*. Nach der Ermittlung der Radiusliste wird eine Schleife ausgeführt, um eine Serie konzentrischer Kreise mit einem Mal zu erstellen. Folgendes Programm enthält die Schleife zum Zeichnen der Kreise.

--------------------------------------- drawgraf.c --

```
/*  Kreis ueber Radiusliste oder Kreisbogen ueber Radiusliste  */
if (npkt > 3  )
{xtemp [0] = xx[0];
  ytemp [0] = yy[0];
  xtemp [2] = xx[2];
  ytemp [2] = yy[2];
  for (i = 0;  i<npkt - 4;  i++ )
        { switch ( ista[2] )
              { case 1 :   /*  Vollkreis ueber Radiusliste ( ungefuellt/gefuellt ) */
                    if ( ista[1] == 1 )
                        gzkre_( xx, yy, &xx[i+3], &yy[i+3], &xx[i+3], &yy[i+3] );
                      else
                       gzfkre_( xx, yy, &xx[i+3], &yy[i+3], &xx[i+3], &yy[i+3], &eins );  break;
              case 5:  /*  Kreisbogen ueber Radiusliste ( ungefuellt/gefuellt ) */
                    xtemp [1] = xx[i+3];
                    ytemp [1] = yy[i+3];
                    if ( ista[1] == 1 ) ista[0] = 1;
                    modff = 0;
                if ( ista[1] >= 3 ) { modff = ista[1] - 2; ista[0] = 1; }/* --> nur fuer > 0 gefuellt */
                  ie_arc_(  &npktneu, xtemp, ytemp, &ista[3], rsta, &ista[4], &rsta[2], &ista[0],
```

```
                &modff, &ifnr );
            break; } /* ENDE switch */
    }
}
```

Bild 4.4-25 zeigt eine Serie konzentrischer Kreise, die durch *Kreis über Radiusliste* erzeugt wurden. Die Generierungsart *Kreisbogen über Radiusliste* funktioniert gleich wie *Kreis über Radiusliste*. Benötig werden hier im mathematisch positiven Drehsinn noch zwei Punkte zur Definition des Anfangspunkts und des Endpunkts der Kreisbögen, siehe Bild 4.4-26.

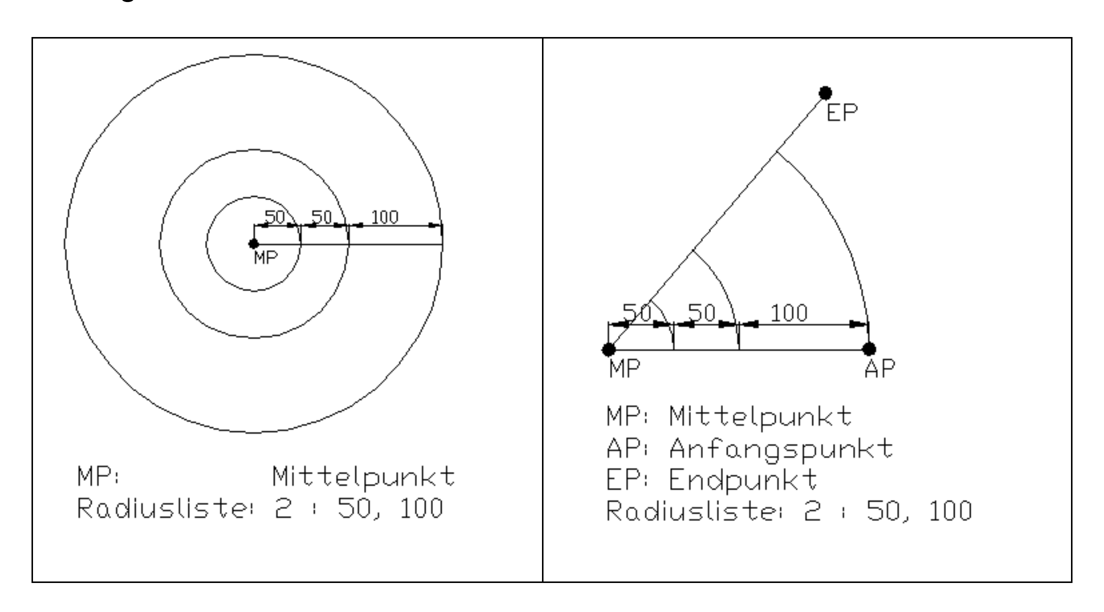


Bild 4.25 Kreis über Radiusliste Bild 4.26 Kreisbogen über Radiusliste

Für die hier neu konzipierte Leistung „Kreis über 3 Punkte" sind die folgende Schritte zu organisieren:

· Anfordern der Eingabe von 3 Punkten

Durch das zum Aufbau und zur Pflege der Konfigurationsdatei dienende Programm **parsereq** wird es möglich, bei der gleichen Aktion vom Benutzer gemäß gewähltem Modus unterschiedliche Eingaben anzufordern.

-- grafik.rqi ---

```
/* ##########
/* GENERIEREN: Kreis
/* ##########
/* -->> über drei Punkte
Begin:65006514,23510000;
 BeginCallback:1,BoPrePowerGrf;
 RechteMausCallback:1,BoCallWasnach;
ShiftRechteMausCallback:
1,BoCreateCombiBox<1500:12351:1505>;
CtrlRechteMausCallback:
1,BoCreateCombiBox<1500:12351:1509>;
BeginReq;
   ReguesType:101;
   CursorType:102;
   MinElReq:1;
   MessFileNr:3500;
   MessNr:58;
   PickMode:25604;
   Securitylevel:-2;
   ErgUsnr:1;
   ErgUstp:18069900;
   PrevUsnr:65140004;
   PrevUstp:10089900;
   CoorSource:10;
   CompareVal:4;
   CompareAdr:1 12351 graf bas2 default 88
4;
   Callback:1,BoSetAnkerRubber<1>;
EndReq;
BeginReq;
    RequesType:101;
   CursorType:2006;
   MinElReq:1;
   MessFileNr:3500;
   MessNr:59;
   PickMode:25604;
   Securitylevel:-2;
   ErgUsnr:2;
   ErgUstp:18069900;
   PrevUsnr:65140004;
   PrevUstp:10089901;
   CoorSource:10;
   CompareVal:4;
   CompareAdr:1 12351 graf bas2 default 88
4;
   PCallback:1,ClbFileSelectInd<1:0>;
 EndReq;
 BeginReq;
   RequesType:101;
   CursorType:2006;
   MinElReq:1;
   MessFileNr:3500;
   MessNr:60;
   PickMode:25604;
   Securitylevel:-2;
   ErgUsnr:3;
   ErgUstp:18069900;
   PrevUsnr:65140004;
   PrevUstp:10089902;
   CoorSource:10;
   CompareVal:4;
CompareAdr:1 12351 graf bas2 default 88 4;
Callback:1,Erg_JoinLists<99:1:2:3>
                /GenKreis<3:99>;
    PCallback:1,ClbFileSelectInd<1:0>;
 EndReq;
/*
```

· Speichern und Abholen der Punktkoordinaten

Die Information der gepickten 3 Punkte wird in eine Liste mit entsprechendem Index gespeichert. Dann wird die Information in *grafdo.c* abgeholt und ggf. die Fehler behandelt.

```
---------------------------------------- grafdo.c ------------------------------------------------------

/* --- HOLE: Koordinate --- */

if ( (status=GR_Erg_GetIdsU(pkt_usnr, &idelem, &id)) ) goto ret;

speic_nel_ ( &id, &dtyp, &ierr, &nel );

if ( BoSetFfeinCstErr(ierr) ){ status=5000000+ierr; goto ret; }

----------------------------------------------------------------------------------------------------
```

· Ermittlung des Kreismittelpunkts

Die Kernfunktion für das Zeichnen eines Kreises oder eines Kreisbogens lautet:

*void gzkre_ (float *x0, float *yy0, float *x1, float *yy1, float *x2, float *y2).*

Die Parameter dieser Funktion sind die Koordinaten des Kreismittelpunkts, des Anfangspunkts und des Endpunkts auf dem Umfang (für Kreise sind Endpunkte gleich Anfangspunkte). Die folgende Routine dient zur Ermittlung des Kreismittelpunkts aus 3 Umfangspunkten (Bild 4-28):

```
--------------------------------------- ge_kreispkt.c ------------------------------------------

void ge_kreispkt(int npkt, float xx[], float yy[], int *kreis)

/*          --Ermittlung Kreismittelpunkt von 3 Punkten auf einem Kreis ---        */

/*                    3 Punkte  xx[0,1,2] ,yy[0,1,2] auf einem Kreis               */

/*                    ➔ Mittelpunkt des Kreises   xx[0],yy[0]                      */

/*  int npkt           ------ Zahl der Punkte                                      */

/*  float xx[], yy[]    ------Punkte auf dem Kreis                                 */

/*  int *kreis          ------ 0 Kein Kreis 1 Kreismittelpunkt ermittelt           */

{float min=0.00001, k1,k2,b1,b2;

 float mx,my;

 if (npkt != 3) {*kreis = 0; return;}

 /* Überprüfen und Ermitteln */

 if ( fabs(yy[1]-yy[0]) < min )

   {if ( fabs(yy[2]-yy[1]) < min ){*kreis = 0; return;}

    else{ mx= (xx[0]+xx[1])/2;

          k2= (xx[1]-xx[2])/(yy[2]-yy[1]);
```

Bild 4.27 Ermittlung Kreismittelpunkt

```
        b2= (yy[2]+yy[1])/2 - k2 * (xx[2]+xx[1])/2;
        my = k2 * mx + b2;
      }
   }
 else{if ( fabs(yy[2]-yy[1]) < min )
        {mx= (xx[2]+xx[1])/2;
         k1= (xx[0]-xx[1])/(yy[1]-yy[0]);
         b1= (yy[0]+yy[1])/2 - k1 * (xx[0]+xx[1])/2;
         my = k1 * mx + b1;
        }
    else{k1=(xx[0]-xx[1])/(yy[1]-yy[0]);
        k2=(xx[1]-xx[2])/(yy[2]-yy[1]);
        b1= (yy[0]+yy[1])/2 - k1 * (xx[0]+xx[1])/2;
        b2= (yy[2]+yy[1])/2 - k2 * (xx[2]+xx[1])/2;
        if ( fabs(k2-k1) < min ){*kreis = 0; return;}
        mx = (b2 - b1)/(k1 - k2);
        my = k1*mx + b1;
      }
   }
xx[0] = mx;
yy[0] = my;
}
```

·Zeichen des Kreises

Nach der Ermittlung des Kreismittelpunkts und des Radius wird die Funktion für Kreiszeichen aufgerufen. Siehe Bild 4.-28

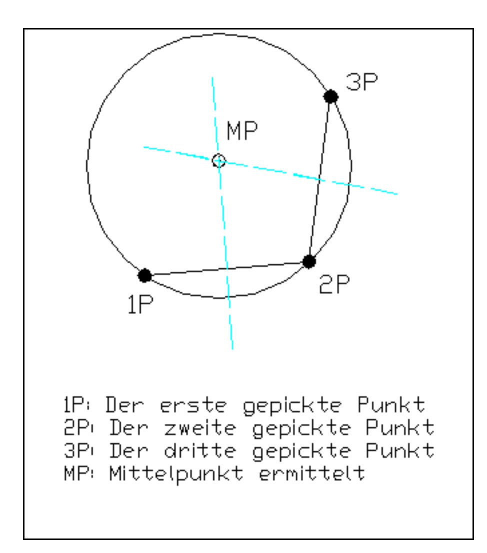


Bild 4.28 Kreis über 3 Punkte

4.4.2 Mehrfach Kopieren über Maße und Winkel

Eine in der Baupraxis häufig genutzte CAD-Leistung ist das Kopieren von 2D- und 3D-Objekten. Folgende Optionen für „Kopieren“ wurden von dem zu Testzwecken zur Verfügung gestellten CAD-System angeboten:

- Kopieren über Punkte

  Eine Kopie wird interaktiv nach Picken des Start- und Endpunkts des Verschiebevektors erstellt.

- Kopieren über Maße

  Eine Kopie wird durch Maßangabe zum Verschiebevektor über ein Eingabefeld erstellt.

Die Analyse der Kopie-Leistung im untersuchten CAD System zeigt offenbar Leistungslücken, die Anwender mit über den Stahlbau hinausgehenden Anforderungen des Metallbaus wie folgt beschreiben:

„Das Kopieren von 2D-Grafik soll über mehrere Maß- oder Winkelangaben auch mehrfach in einer Aktion möglich sein. 2D-Kopieren bietet bisher kein Eingabefeld für Maß- und Winkelangaben.“

Bei der Realisierung der Anforderung Mehrfach Kopieren über Maße wird zunächst die Oberfläche bzw. das Dialogfenster entworfen. Bild 4-29 zeigt die neu konzipierte Dialogbox für „Grafik/Kopieren über Maße“.

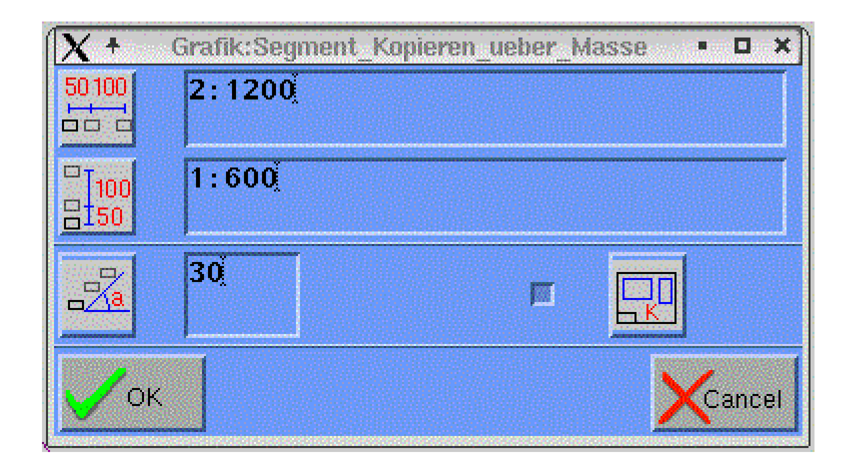


Bild 4-29 Eingabefenster für Kopieren über Maße

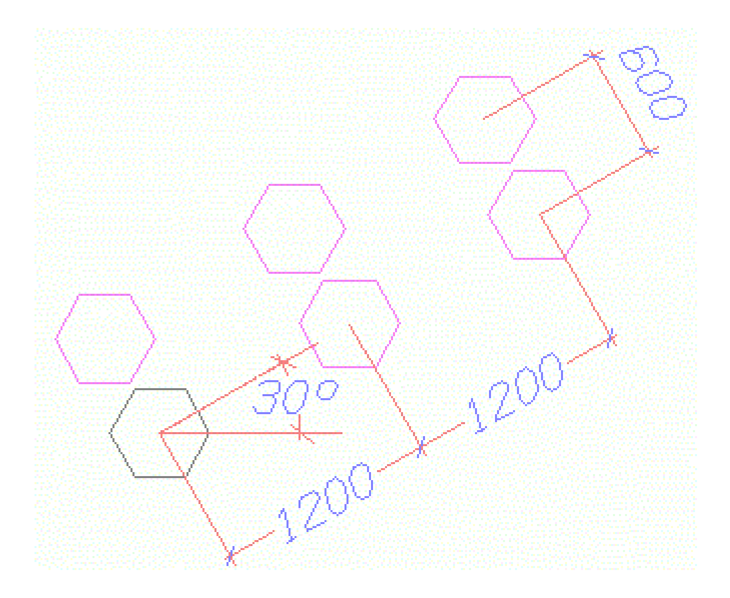


Bild 4-30 Mehrfach Kopieren über Maße

Das Ergebnis der Eingabedaten in Bild 4-29 zeigt Bild 4-30 an einem Beispiel.
Im Eingabefeld „**in X**", „**in Y**" und „**Drehwinkel**" werden folgende Angaben dem System mitgeteilt:

- Anzahl der Kopien und ihr relativer Abstand zueinander in X-Richtung
- Anzahl der Kopien und ihr relativer Abstand zueinander in Y-Richtung
- Drehwinkel der beabsichtigten Rasterung zur X-Richtung

Nach der Identifizierung durch Picken der Objekte, die mehrfach über Maße kopiert werden sollen, wird die Information des Objektes, Koordinaten und Index, in der Datenbank gesucht. Die Kernfunktion für das Kopieren ist die gleiche Funktion wie für Verschieben und Rotieren, jedoch wird das Original-Objekt in der Datenbank nicht gelöscht. Eine Schleife wird ausgeführt, um mehrere Kopien zu erstellen. Bild 4-30 zeigt ein Beispiel für *Kopieren über Maße* mit der Eingabe, die Bild 4-29 zeigt, nämlich **in X** 2:1200, **in Y** 1:600 und **Drehwinkel** 30. Die Eingaben 2:1200, 2000, 3000 sind auch möglich.
Beim *Mehrfach Kopieren über Winkel* werden mehrere Kopien erstellt, die um einen wählbaren Punkt rotiert werden. Eine Möglichkeit für die Eingabe des Drehwinkels sollte zunächst vom System nach der Durchführung des Befehls angeboten werden. Bild 4-31 zeigt diese Eingabemöglichkeit für *Mehrfach Kopieren über Winkel*. Im

Eingabefeld ist entweder der Rotationswinkel zueinander mit Komma wie ***60, 30, 90*** oder die Anzahl der Kopien (Replikator) und ihr Drehwinkel wie ***2:60, 30*** einzugeben. Auch Kombinationen sind erlaubt.

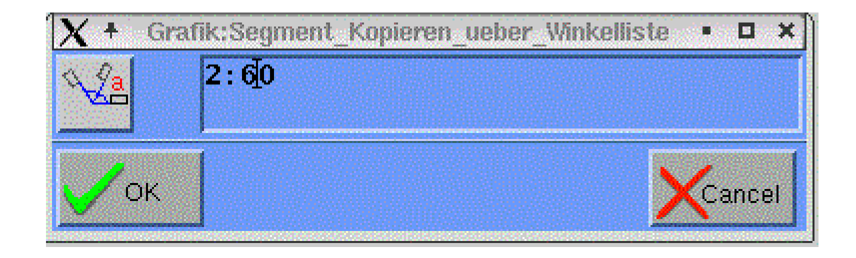


Bild 4-31 Eingabefenster für Mehrfach Kopieren über Winkel

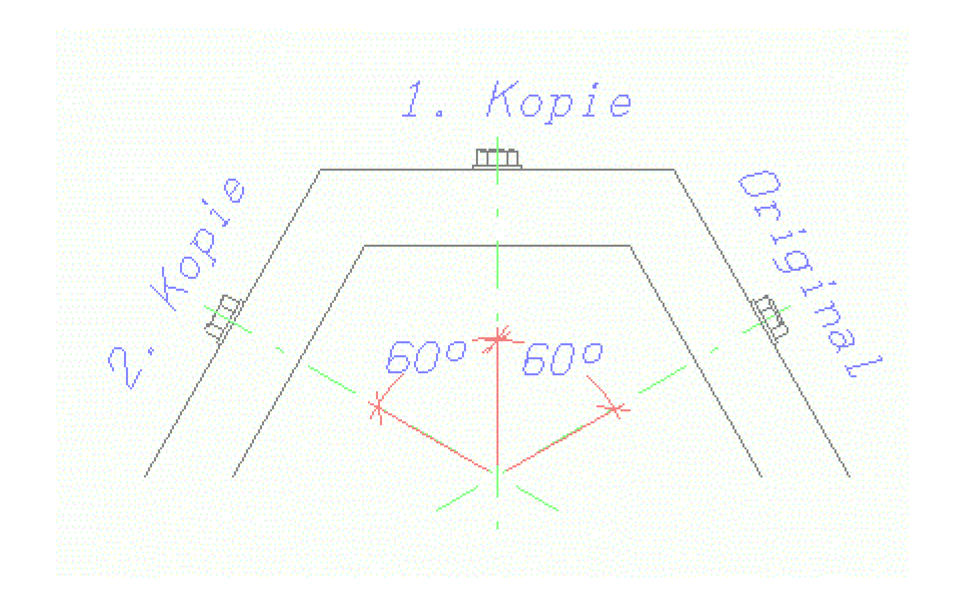


Bild 4-32 Mehrfach Kopieren über Winkel

Ein Beispiel der Daten des Bildes 4-31 zeigt Bild 4-32.

Nach der Identifizierung der Objekte, die mehrfach über den eingegebenen Winkel gedreht kopiert werden, muss ein Bezugspunkt, von dem die Drehung gestartet werden soll, gepickt werden. Die Information des identifizierten Objektes, Koordinaten, Index und Objekttyp, wird in der Datenbank gespeichert, siehe folgenden Code im Bearbeitungsprogramm *transelem.c*:

```
---------------------------------------- transelem.c ----------------------------------------------------
/* --- Elementliste --- */
if ( (status=GR_Erg_GetIdLSEU(usnr_erg1, &idelem) )) goto weiter;
```

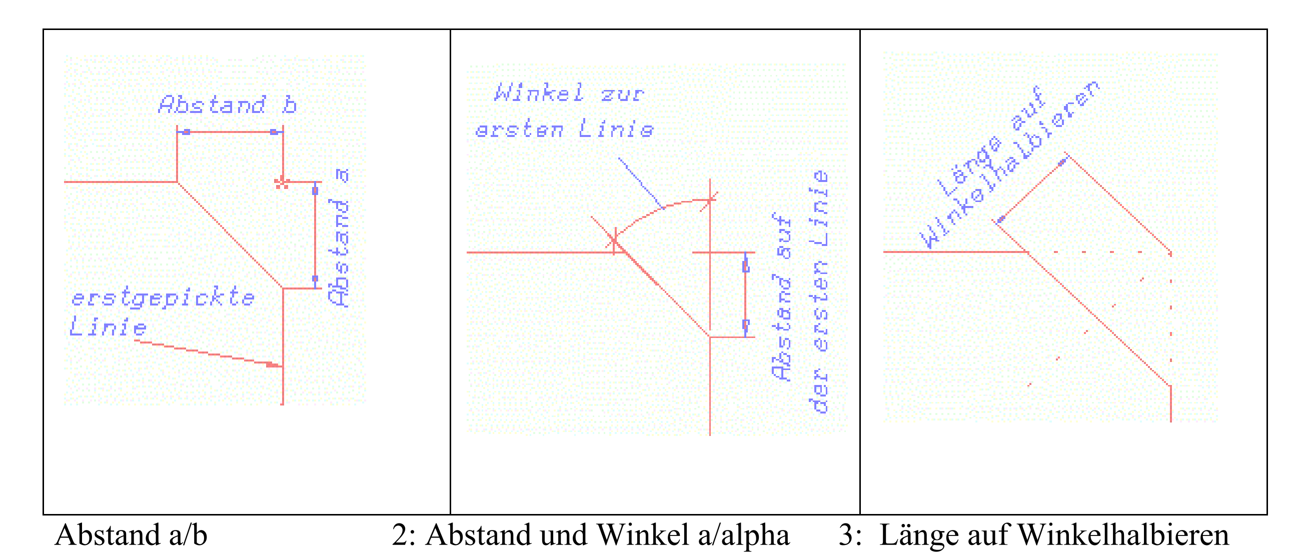
```
speic_nel_ ( &idelem, &Itypelem, &ierr, &nelem );
if ( ierr || nelem < 1 )
 { BoGetSelTextUser ( 2, 12351, 6500, 18, sizeof(scratch), scratch );
  /* Fehler bei der Punktselektion */
  BoModalError ( scratch );
       goto weiter;
 }
elem = (O_LSE *)speic_adr ( &idelem );
------------------------------------------------------------------------------------------------------------
```

Danach werden die neuen Objekte, d. h. die Kopien der ausgewählten Objekte, in der Datenbank eingetragen. Nach der Transformation gemäß des Drehwinkels werden die Objekte neu platziert. Schließlich werden die Kopien in der aktuellen Zeichnung dargestellt.

Bild 4-31 zeigt ein Beispiel für *Kopieren über Winkel* mit der Eingabe, die Bild 4-32 zeigt, nämlich **Winkel** 2:60.

4.4.3 Linien fasen und abrunden

Häufig müssen zwei Linien, die sich schneiden oder sich in ihrer Verlängerung schneiden würden, mit einem zuvor festgelegten Abstand oder Radius verbunden werden (Bild 4-33). Diese Leistung fehlt dem untersuchten CAD-System.



Abstand a/b 2: Abstand und Winkel a/alpha 3: Länge auf Winkelhalbieren

Bild 4-33 Generierungsarten *für Linien fasen*

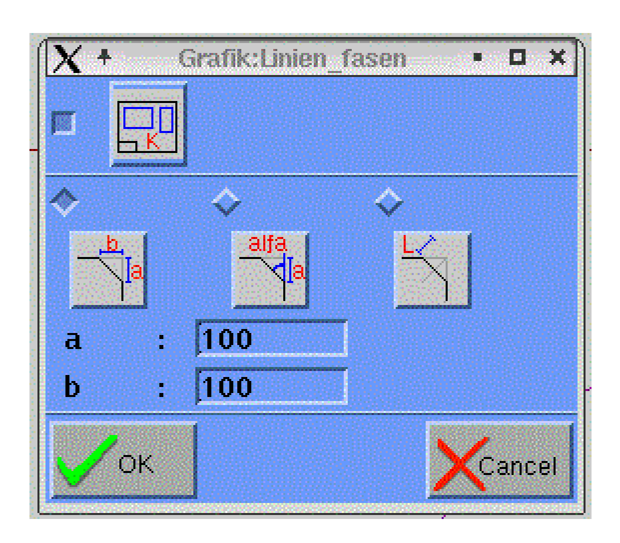


Bild 4-34 Eingabefenster für *Linien fasen*

Zur Entwicklung der Leistung „*Linien fasen*“ wird zunächst das Dialogfenster für das Festlegen der Fasungsart entworfen. Bild 4-34 zeigt das konzipierte Dialogfenster für "Grafik/Linien fasen". Im Listenfeld *Generierungsart* werden 3 Möglichkeiten angeboten, nämlich:

- Abstand a/b: Fase wird durch zwei Abstände festgelegt.
- Abstand und Winkel a/alpha: Fase wird durch einen Abstand auf der ersten Linie und einen Winkel zur ersten Linie festgelegt.
- Länge auf Winkelhalbierende: Fase wird durch eine Länge auf der Linie der Winkelhalbierenden festgelegt.

Bild 4-33 zeigt diese drei Generierungsarten. Nach der Auswahl der Generierungsart erscheint ein der Art entsprechendes Eingabefeld. Nachdem der Anwender z. B. „Abstand a/b“ ausgewählt hat, erscheinen zwei Eingabefelder für Abstand a und Abstand b.

Nach der Identifizierung zweier Linien wird berechnet, ob und um welche Werte sie gedehnt oder gestutzt werden müssen. Schließlich werden die zwei identifizierten Linien durch die Fasungslinie verbunden.

Beim *Linien Abrunden* werden zwei Linien durch einen genau eingesetzten Bogen mit festgelegtem Radius verbunden. In diesem Fall ist nur ein Radius einzugeben, siehe Bild 4-35 für die Eingabe und Bild 4-36 für das graphische Ergebnis.

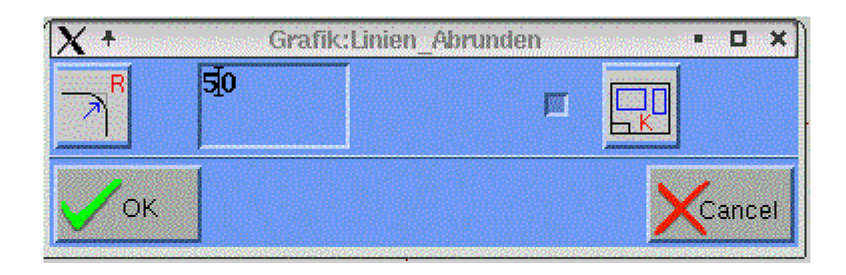


Bild 4-35 Eingabefeld für *Linien abrunden*

Nach der Identifizierung zweier Linien wird berechnet, ob und um welche Werte sie gedehnt oder gestutzt werden müssen, damit sie mit einem Ausrundungsbogen tangential verbunden werden.

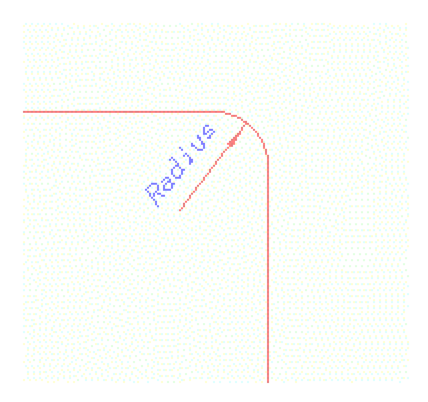


Bild 4-36 Zwei Linien mit Radius abgerundet

4.4.4 Verketten von Primitiva und Segment auflösen

Eine in der CAD-Konstruktionspraxis gebrauchte Leistung zur Verwaltung von Objekten in Zeichnungen ist die Verkettung bzw. Gruppierung mehrerer Grafikelemente zu einer einzigen Komponente, Segment genannt. Umgekehrt ist auch ein solches Grafiksegment in seine Komponenten zu zerlegen. Anwender definieren diese Leistung folgendermaßen:

„**Verketten** von N Primitiva, z. B. mehreren Linien, zu einem Segment und die umgekehrte Aktion **Auflösen** eines Segments in seine Primitiva wird gewünscht “.

Segment und Primitivum sind zwei verschiedene Stufen der Grafikelemente des untersuchten Systems. Es besitzt eine eigene Grafikschnittstelle, das so genannte bmf_-Format. Diese Schnittstelle besteht hauptsächlich aus den Einheiten »Gruppe«, »Segment« und »Primitivum«. Jede Zeichnung besitzt eine Rootgruppe, auf der die anderen Gruppen mit einer Baum-Struktur aufbauen. Jede Gruppe besteht aus Segmenten, beispielsweise Linien, Polygon, Text, Bemaßungsobjekten usw. Jedes Segment kann aus mehreren Primitiva bestehen. Primitiva sind die kleinste Grafikeinheit. Bild 4-37 zeigt die Struktur einer Grafikdatei im Format bmf_. Der Name „bmf_“ leitet sich her aus BOCAD Meta File.

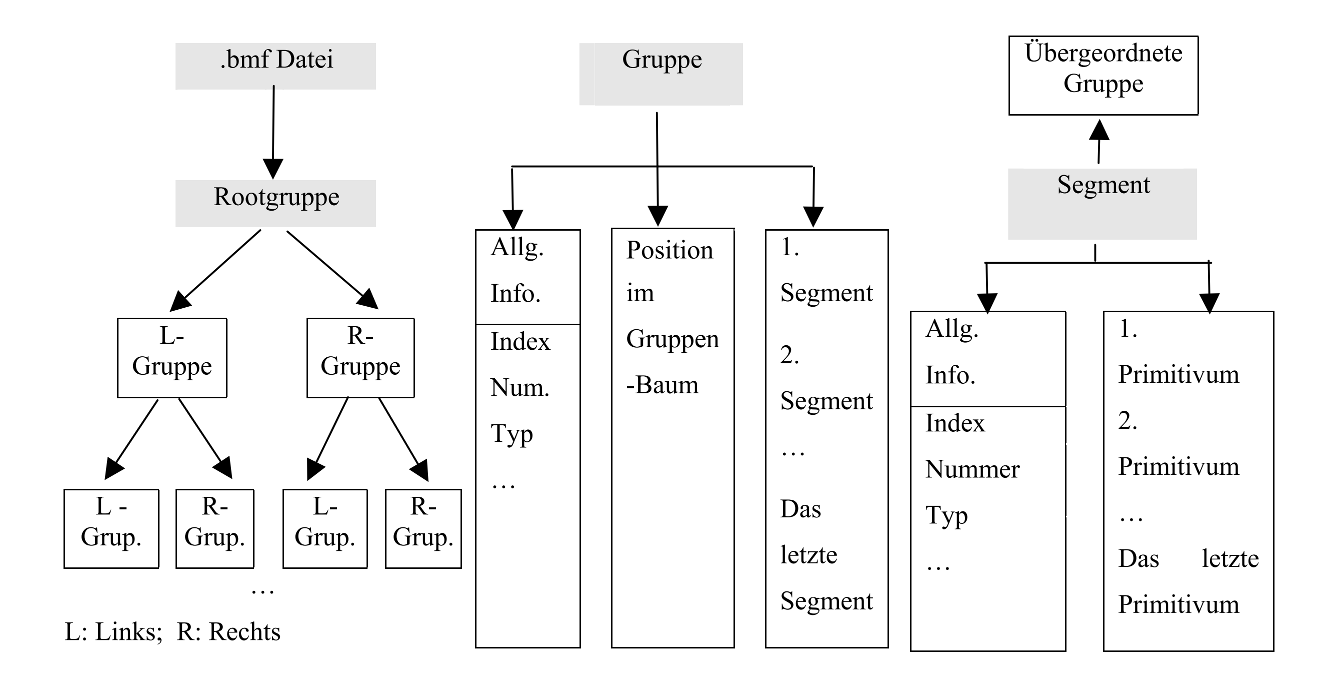


Bild 4-37 Struktur einer Grafikdatei bmf_ Format

Bei der programmtechnischen Realisierung der Anforderung „*Verketten von N Primitiva*“ werden im ersten Schritt die vom Benutzer identifizierten Primitiva in der Datenbank vom bisherigen, übergeordneten Segment gelöst. Dann wird ein neues Segment erstellt, das aus den identifizierten Primitiva besteht. In der dazugehörigen Gruppe wird das Segment eingetragen. Schließlich wird das neue Segment gezeichnet.

Bei der Leistung „*Segment auflösen*“ werden Segment-Objekte in ihre einzelnen Bestandteile zerlegt, ohne dass dies graphisch sichtbar wird. *„Segment auflösen“*

macht beispielsweise aus einer Maßkette die Primitiva Linien, Bogen und Text. Bei der Entwicklung „*Segment auflösen*" sind folgende Schritte durchzuführen:

- Die Information der von dem Benutzer identifizierten Segmente werden in der Datenbank abgeholt.
- Alle Primitiva, die zum identifizierten Segment gehören, werden durchsucht. Jedes Primitivum wird zu einem neuen Segment mit dem entsprechenden Segmenttyp gewandelt.
- Die vom Benutzer identifizierten Segmente werden mit einer Kennung markiert. Die neuen generierten Segmente werden in der zugehörigen Gruppe eingetragen.
- Die Zeichnung wird erneut dargestellt.

Der folgende Code zeigt die Realisierung der Leistung *„Segment auflösen"*.

-- transelem.c --

```
/* ------------------------------------------------------------------------------ */
void BoAuflSegExe ( Widget w, XtPointer client, XtPointer call )     /* Y.Chang */
/* ------------------------------------------------------------------------------ */
{int   i, j, idolse=0, itype=21,i22 = 22, nelem=0, status=0, ifnr,
       aktfile, segind_quel, segind_neu, aseg, dbelem,
       vpind, allprim=123456, delori, lel, primtyp,
       elem_usnr, eins=1;
 int   segtyp, deltyp, segnr, ind_ziel;
 int   ifile_quel, ifile_ziel,grpind_quel, grpind_ziel, scalieren = 0;
 double didvgl,scal[2]={1.0,1.0}, theta=0;
 double xgrup, ygrup, q_scal[2], q_theta_blatt,z_theta_blatt, q_trala[2],
       scal_dum[2]={1.0,1.0}, z_trala[2],
       rot_dum[2]={0.0,0.0},theta_dum=0,z_scal[2]={1.0,1.0} ;
 O_LSE  *elem, seg;
 O_DKOO refe = {0.0,0.0,0.0},*bezpkt;
 char   scratch[128];
 char printf_start[80] = " *** Start BoAuflSegExe ***\n";
 char printf_ende[80]  = " ***  Ende BoAuflSegExe ***\n";
 char printf_err[80]   = " ***  >>>> BoAuflSegExe, status: %d ***\n";
```

```
if ( GrTrafTestLv ) printf (printf_start);
(void) BoGrfUndoStart((char *) NULL,3, 67);

BoAppParmText ( client, "BoAuflSegExe", scratch );
ApParms(":, ",scratch, 2,&delori, 2,&elem_usnr, 0);
/* HOLE: Kopier-Elemente */
if ( (status=GR_Erg_GetIdLSEU(elem_usnr, &idolse)) ) goto cancel;
if ( Check_Grp(idolse) ) { status=12345;  goto cancel; }
if ( !scalieren )
 { X_ScalFak= 1.0;
   Y_ScalFak= 1.0;
 }

if (GrTrafTestLv>1)
 { printf(" ---- O_LSE <Segmente> --------\n");
   {int nd=1, des[3]={2,0,3}; DumpId(idolse,nd,des); }
   printf(" -----------------------------\n");
 }
speic_nel_(&idolse, &itype, &status, &nelem);
if (status) { status+=500000; goto cancel; }

elem = (O_LSE *)speic_adr ( &idolse );

/* --- Keine Db-Elemente zugelassen --- */
dbelem = InqGrfInDbErr ( idolse, 2);
if ( dbelem >0 ) { status=12345; goto cancel; }

aktfile     = own_is_akt_file_ ( );
ifile_quel  = BoGetPickFileInd (idolse,0);      if ( ifile_quel<0  ) { status=12345; goto cancel; }
ifile_ziel  = ifile_quel;
grpind_quel = BoGetPickGrupInd (idolse,0);      if ( grpind_quel<0 ) { status=12345; goto cancel; }
grpind_ziel = grpind_quel;

/* --- Viewportnummer --- */
GetNrSlotByOlse ( &idolse, &vpind, &status);
if (status) { status+=510000; goto cancel; }

if ( GrTrafTestLv > 2)
```

```
{ printf(" >>>> ----------------------------- <<<<\n");
  printf(" >>>>        Dump_Gruppe          <<<<\n");
  printf(" >>>> ----------------------------- <<<<\n");
  {int nwas=0, dump[29];
   own_dump_ginfi_( &nwas, dump, &grpind_quel); }
 }

bezpkt = &refe;
/* ------------------ */
/* >>> AUSFUEHREN <<< */
/* ------------------ */
aseg   = own_max_seg_ ();
own_kop_prat_( &ifile_quel );
for (i=0; i<nelem; i++, elem++)
  {seg.segind  = elem->segind;
   seg.grpind  = elem->grpind;
   seg.primind = elem->primind;
   own_st_akt_  ( &ifile_quel );
   segind_quel = BoGetPickSegInd  (idolse,i);
       own_reo_segi_(&seg.segind, &ifnr);
       if (ifnr!=1) { status=600000+ifnr; goto cancel; }
       own_mpur_sinfi_ (&didvgl, &eins, &i22,&(seg.segind));
       lel = (int)didvgl;
       own_cls_seg_( );
       for (j=0;  j<lel;  j++ )
       {own_reo_segi_(&seg.segind, &ifnr);
           (void *) Own_mpur_prim (&didvgl, &eins, &eins, &j);
       primtyp = (int)didvgl ;
       own_cls_seg_( );
           own_kop_prim_ (&ifile_quel, &segind_quel, &j, &ind_ziel );
       switch ( primtyp )
              {case   2: /* polylines, Segmenttyp--9139100*/
                   case  16: /* Fill area, Segmenttyp--9139100*/
                   case  22: /* polylines, Segmenttyp--9139100*/
                         /*  Segment bereitstellen                     */
                          segtyp      = 9139100;
                                      break;
               case   7: /* Kreisstueck, Segmenttyp--9199100*/
                   case  19: /* Kreisstueck gefuellt, Segmenttyp--9199100*/
```

```
                    /*  Segment bereitstellen                   */
                     segtyp      = 9199100;
                       break;
            case  4: /* text, Segmenttyp--9229300*/
                    /*  Segment bereitstellen                   */
                     segtyp      = 9229300;
                              break;
            default: /* wiessnicht */
                              break;
        }
    OwnSetSegType (ind_ziel, segtyp);
     /* ---------------------------------------------- */
     /* QUEL-Segment in den Bezugspunkt Transformieren */
     /* ---------------------------------------------- */
      /* HOLE: Verschiebungsvektor           */
     /*      Gruppennullpunkt -> Bezugspunkt */
  GetGrpBlatInf ( grpind_quel, &xgrup, &ygrup, &q_scal[0], &q_scal[1], &q_theta_blatt, &status );
   if (status) goto cancel;
    GetGrpKonVekL ( grpind_quel, &xgrup, &ygrup, &bezpkt->x, &bezpkt->y, &q_trala[0], &q_trala[1],
  &status );
    if (status) goto cancel;
  /* TRANSFORMIEREN: >>> Bezugspunkt -> Gruppennullpunkt <<< */
   own_reo_segi_ ( &ind_ziel, &ifnr );
   if (ifnr!=1) { status=600000+ifnr; goto cancel; }
   q_trala[0] *=-1;
   q_trala[1] *=-1;
   bmf_pr_trans_( &ifile_quel, &ind_ziel, &allprim, scal_dum, q_trala, rot_dum, &theta_dum );
   own_cls_seg_nochange ( );  /* Segment schließen und Gruppenextrema nicht behandeln */
   /* --------------------- */
   /*  QUEL-Segment kopieen */
   /* --------------------- */
  own_st_akt_  ( &ifile_ziel );
   /* --- Segment bereitstellen --- */
   aseg   = own_max_seg_ ();
  /* --- Gruppe öffnen --- */
   CShoSetAktVpGrp ( vpind );

  own_kop_prat_  ( &ifile_quel );
   own_kop_segnv_ ( &ifile_quel, &ind_ziel, &segind_neu );
```

```
/* --- Gruppe wieder schließen --- */
/*CShoDetAktVpGrp ( vpind );*/
/* Segment zur potentiellen Pickliste zugefuegt */
BoAddNewSegs( client, aseg);
/* ------------------------------------------- */
/*  NEU-Segment in den Zielpunkt Transformieren */
/* ------------------------------------------- */
/*own_st_akt_  ( &ifile_ziel );*/
/* HOLE: Verschiebungsvektor           */
/*      Gruppennullpunkt -> Zielpunkt */
/* ---------------------------------- */
GetGrpBlatInf ( grpind_ziel, &xgrup, &ygrup,&z_scal[0], &z_scal[1], &z_theta_blatt, &status );
if (status) goto cancel;
GetGrpKonVekL ( grpind_ziel, &xgrup, &ygrup, &bezpkt->x, &bezpkt->y, &z_trala[0], &z_trala[1],
&status );
if (status) goto cancel;

scal[0]  = ( z_scal[0] / q_scal[0] ) * X_ScalFak;
scal[1]  = ( z_scal[1] / q_scal[1] ) * Y_ScalFak;
theta    = q_theta_blatt - z_theta_blatt;

own_reo_segi_ ( &segind_neu, &ifnr );
if (ifnr!=1) { status+=601000; goto cancel; }
 bmf_pr_trans_( &ifile_ziel, &segind_neu, &allprim, scal, z_trala, rot_dum, &theta );
 own_cls_seg_( );
 if ( delori )
  { own_del_segi ( ind_ziel);
  }
}
/*  LÖSCHE: QUEL-Segment */
   if ( delori )
      {OwnSetSegDelFlag(0);
    own_st_akt_  ( &ifile_quel );
    own_del_segnod_ ( &segind_quel, &segnr, &deltyp);
      }
/* --- Gruppe wieder schließen --- */
CShoDetAktVpGrp ( vpind );
}
/* Gruppenextrema aktualisieren */
```

```
  own_ext_root_( );
  /* ---  Resourcen freigeben --- */
  own_st_akt_ ( &aktfile );   /* current wiederherstellen          */
  status=0;
  cancel:;
  (void) GR_FreeCidRes (&idolse);
  Erg_FreeIdsU(elem_usnr);

  (void) BoGrfUndoEnd(status);
  if ( status && status!=12345 ) printf (printf_err,status);
  if ( GrTrafTestLv) printf (printf_ende);
}
```

4.4.5 Linienzug

Bei dem untersuchten CAD-System konnte der Anwender bisher mit dem Befehl Linie-Generieren nur eine einzelne Linie zeichnen. In der CAD-Konstruktionspraxis wird aber gewünscht, dass mehrere Linien hintereinander gezeichnet werden können, und zwar so, dass der Anfangspunkt der neu zu zeichnenden Linie der Endpunkt der zuletzt gezeichneten Linie ist. Diese Entwicklungsaufgabe ist ein überschaubarer Fall für ein intelligentes 2D-Objekt.

Die bisherige Funktion für **Linie Generieren** wird daher analysiert. Die Kernfunktion dafür lautet:

void GenLine (Widget w, XtPointer client, XtPointer call)

In dieser Funktion werden die von dem Benutzer gepickten Punkte behandelt. Theoretisch kann die Zahl der gepickten Punkten 0, 1, 2 oder mehrere sein. Bei einer Zahl gleich 0 oder 1 wird keine Linie erzeugt. Falls 2 Punkte gepickt wurden, wird eine Linie generiert bzw. gezeichnet. Wenn mehr als 2 Punkte gepickt wurden, sollte entsprechend der Punktfolge ein Polygon gezeichnet werden. Als problematisch wird der vorgefundene Lösungsansatz erkannt angesehen, dass das Werkzeug **Parsereq** für den Pickvorgang die zwei Parameter für „Linie Generieren", MinElReq und MaxElReq, jeweils zu 2 einstellt. MinElReq definiert dabei die minimale Anzahl an Punkten, die gepickt werden müssen, um den Request „**Linie Generieren"** zu befriedigen. MaxElReq definiert hingegen die maximale Anzahl an

Punkten für den Request „**Linie Generieren“**. Wegen dieses unglücklichen, bisherigen Lösungsansatzes muss der Benutzer bei „**Linie Generieren“** für jedes Linienstück immer zwei Punkte picken. Wenn eine Linienfolge erzeugt wird, muss der Befehl komplett wiederholt werden, d. h., der Benutzer muss immer den Anfangspunkt der neu zu generierenden Linie und den Endpunkt picken. Als „intelligente Leistung“ wird aber, dass der Anfangspunkt der neu zu generierenden Linie der Endpunkt der zuletzt zu generierten Linie ist.

Um diesen neuen Lösungsansatz zu verwirklichen, wird der Parameter MaxElReq, der die maximale Zahl der zu pickenden Punkte definiert, nicht auf zwei beschränkt, sondern bleibt frei.

Zusätzlich sollte sich der Cursor in diesem Fall gummibandartig mitbewegen. Das heißt, wenn der Benutzer nach dem Picken eines Punktes den Mauszeiger bewegt, wird eine Hilfslinie als „Gummiband“ angezeigt und mitbewegt. Diese Hilfslinie zeigt vorübergehend eine Verbindung zwischen dem aktuellen Mauszeigerort und dem Endpunkt der zuletzt gezeichneten Linie.

Um diese „intelligente" Hilfslinie auf dem Bildschirm darzustellen, werden die folgenden drei Schritte durchgeführt (siehe auch den folgenden Code in *grafik.rqi*):

```
----------------------- grafik.rqi -----------------------------------
Begin:65006610,23510000;
 BeginCallback:1,BoPrePowerGrf;
 RechteMausCallback:1,BoCallWasnach;
 ShiftRechteMausCallback:1,BoCreateCombiBox<1500:12351:1505>;
 CtrlRechteMausCallback:1,BoCreateCombiBox<1500:12351:1509>;
 BeginReq;
  ReguesType:101;
  CursorType:102;
  C2:2103;
  MinElReq:2;
  MessFileNr:3500;
  MessNr:-1;
  Messages:1,2,2;
  PickMode:25604;
  Securitylevel:-2;
  ErgUsnr:1;
  ErgUstp:18069900;
  LinieEcho: 101
  PrevUsnr:66100000;
  PrevUstp:20899000;
  CoorSource:1;
  CompareVal:1;
  CompareAdr:1 12351 graf bas2 default 40 4;
  Callback:1,GenLine<1:1:-1>;
 EndReq;
```

- Der Parameter *LineEcho* für die 2-Punkt-Form **„Linie Generieren"** wird in grafik.rqi dazu auf 101 eingestellt. *LineEcho* dient dazu, dass beim Koordinatenpicken 2 aufeinanderfolgende Punkte durch eine Linie verbunden werden. Ob die Verbindungslinie temporär oder permanent ist, steuert Vorzeichen („+" oder „-").

  LineEcho = Vorzeichen (Linientyp * 100 + Farbe)

- Der Parameter *gonote* in *CursorType* in grafik.rqi ist auf 2 einzustellen.

  CursorType: Art des Cursors (Zeigers) als gepacktes Wort

  = grid * 100 000 + warp * 10 000 + gonote * 1 000 + anker *100 + fix * 10 + drag.

  gonote: wie auf gemerkte Position gehen

  =0, nicht gehen

  =1, gehen, dabei Zeiger direkt bewegen

  =2, gehen, nach nächster Mausbewegung, um eine Hilfslinie als "Gummiband" anzuzeigen

- Die Hilfslinie muss mit dem zuletzt gepickten Punkt beginnen. Dieser Punkt muss jeweils in *BoReqStoreSteigung2d (x0, y0, x1, y1)* im Callback *GenLine* als (*x0, y0*) eingestellt werden, sonst würde sich die Hilfslinie immer auf den ersten gepickten Punkt beziehen. (*x1, y1*) sind die Koordinaten des aktuell gepickten Punktes.

4.4.5 Ansichtsfenster mit unregelmäßigen Umgrenzungen

Im untersuchten CAD-System kann der Benutzer zwar verschiedene Ansichten erstellen, aber alle Ansichten haben rechteckige Formen. In der CAD-Konstruktionspraxis kann aber gewünscht sein, dass ein Ansichtsfenster nicht rechteckige Umgrenzungen enthält. In AutoCAD 2000 können z. B. Objekte, die im Papierbereich gezeichnet wurden, in ein neues Ansichtsfenster mit einer Polylinie als Rand abgebildet werden. Bei der Definition einer unregelmäßigen Umgrenzung für das Ansichtsfenster berechnet AutoCAD die Grenzen des ausgewählten Objekts, platziert ein Ansichtsfensterobjekt an der Ecke der Grenzen und schneidet das

ausgewählte Objekt passend zu.

Die Kernfunktion für diese Leistung ist Zuschneiden einer Linie und bzw. eines Kreises oder Kreisbogens durch die Umgrenzung.

Zuschneiden einer Linie durch ein Polygon

Jede Linie, die sich mit dem Polygon schneidet, wird durch das Polygon getrimmt. Nur der Linienabschnitt, der sich innerhalb des Polygons befindet, wird dargestellt.

Bei der hier neu durchgeführten Entwicklung werden zunächst die Schnittpunkte der Linie und des Polygons ermittelt. Danach werden diese Schnittpunkte nach den Abständen zum Anfangspunkt der zu bearbeitenden Linie sortiert, so dass sich die Linie in eine Reihe von Strecken zerlegt. Für jede Strecke wird berechnet, ob sie innerhalb des Umgrenzungspolygons liegt. Ist dies der Fall, wird diese Strecke gespeichert und dargestellt.

Zuschneiden eines Kreises (Kreisbogens) durch ein Polygon

Jeder Kreis (Kreisbogen), der sich mit dem Polygon schneidet, wird durch das Polygon getrimmt. Die Kreise oder Kreisbögen, die keine Schnittpunkte mit dem Polygon haben, liegen entweder außerhalb oder innerhalb des Polygons. Wenn ein beliebiger Punkt auf dem Kreis oder dem Kreisbogen sich innerhalb des Polygons befindet, liegt der Kreis (Kreisbogen) vollständig innerhalb des Polygons. Wenn ein beliebiger Punkt auf dem Kreis (Kreisbogen) außerhalb des Polygons ist, liegt der Kreis (Kreisbogen) außerhalb des Polygons.

Bei der Entwicklung sind die folgenden Schritte zu programmieren:

- Jede Linie des Polygons wird durchsucht, um die Schnittpunkte zwischen Kreis und Polygon zu ermitteln.
- Das Schnittpunkt-Array wird gemäß eines Winkels sortiert, der vorher durch die Linie zwischen dem Schnittpunkt und dem Kreismittelpunkt festgelegt wird. Der Kreis (Kreisbogen) wird in eine Reihe von Kreisbögen zerlegt.
- Jeder Kreisbogen wird berechnet, ob er innerhalb des Polygons ist. Alle Kreis-bogen, die sich innerhalb des Polygons befindet, werden in der Grafik-

Datenbank gespeichert und in eine neue Ansicht mit möglicherweise neuem Maßstab dargestellt.

Das Bild 4-38 zeigt das Zuschneiden eines Kreises durch einen Polygon.

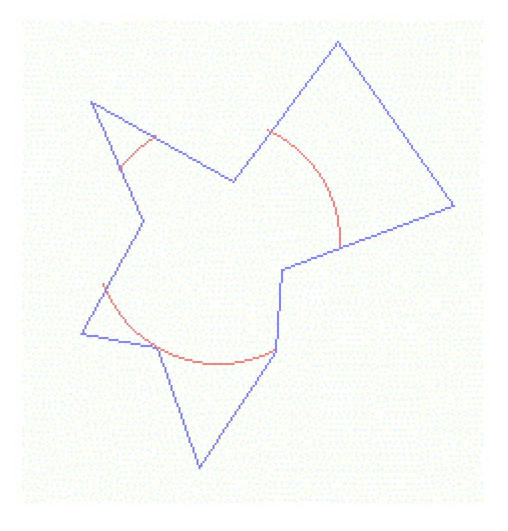


Bild 4-38 Zuschneiden eines Kreises durch einen Polygon

5. Nutzoberfläche und Methoden intelligenter CAD-Objekte

Die theoretischen Grundlagen zur Nutzoberfläche und den Methoden intelligenter Objekte wurden im Abschnitt 4 dargestellt. In den nachfolgenden Abschnitten werden konkrete Ausprägungen für einzelne Objektklassen hergeleitet.

5.1 „Punkte im Raum“

Manche CAD-Systeme nutzen Punkte als eigenständige Objekte kaum. Diese Systeme bilden stattdessen temporär in der Nähe des Mauszeigers an Hilfslinien und Körperkanten pickbare Marker auf Linienenden, Linienschnitten, Linienmitten, Schraubenmitten und ähnlichen Objekten. Die einzelnen Marker haben dabei je nach Bedeutung (Schraubenmitte, Linienschnitt) unterschiedliche, sprechende Formen. CAD-Systeme der oberen Leistungsklasse bieten beide Möglichkeiten wegen ihrer spezifischen Vorteile.

Punkte im Raum können entweder eigenständige Objekte sein oder Elemente eines übergeordneten Objekts, beispielsweise Konturpunkte eines Blechs, Eckpunkte eines Profilstabs, Linienenden oder Linienmitten. In diesem Abschnitt werden eigenständige Punktobjekte behandelt, Punkte als Eckpunkte von Körpern folgen in Abschnitt 5.3.

In dem als Versuchsfeld genutzten CAD-System BOCAD-3D werden Punkte im Raum als eigenständige Objekte intensiv genutzt. Dementsprechend ist die Erzeugung und Modifikation dieser Punktobjekte dort besonders leistungsfähig gestaltet. Nach Wählen des Pickmodus „Teile“ oder „Schrauben“ sind auch in diesem System Schraubenmitten oder Teileecken adäquat zu Punkten wählbar, jedoch ohne temporär in Mauszeigernähe angebotene Marker.

Marker als pickbare Punkte sind bei hinreichend großem Darstellungsmaßstab der Umgebung des Mauszeigers eine effiziente und bei Demonstrationen beeindruckende Lösung. Sie werden ohne Arbeitsaufwand von ohnehin bereits existierenden Objekten angeboten. Laufende Aktionen müssen nicht unterbrochen werden, um fehlende Punkte zu erzeugen. Eigenständige Punktobjekte hingegen

müssen erst in einer eigenen Aktion erzeugt werden, bevor sie nutzbar sind. Dennoch können sie sehr sinnvoll und unter Umständen effizienter sein als temporäre Marker. Bei kleinem Maßstab können nämlich temporäre Marker so dicht benachbart sein, dass leicht versehentlich der falsche Marker gewählt wird, z. B. eine Stegkante statt der Stegmitte eines Profils. Insbesondere bei modernen, dünnwandigen Profilen besteht diese Gefahr. Folge sind dann unauffällige Verschiebungen in Millimeter-Größenordnung bei den weiteren Aktionen, die nicht oder erst spät anhand von Folgefehlern bemerkt werden. Die Rückverfolgung zum ursprünglichen Fehler und die Beseitigung aller Folgefehler ist dann aufwendig. Um diese Gefahr zu verringern, ist häufiges Zoom-in und Zoom-out notwendig, was den Effizienzvorteil kompensieren kann.

Demgegenüber werden Punktobjekte im Pickmodus „Punkte“ immer sicher getroffen, selbst bei starker Überfrachtung des Bildausschnitts. Die Gefahr von dicht benachbarten Punktanhäufungen, deren Punkte ebenfalls erst nach Vergrößerung der Darstellung treffsicher erreichbar wären, ist relativ gering. Der Ingenieur am Bildschirm erzeugt schließlich nicht unnötig überflüssige Punktobjekte, Marker jedoch erscheinen zur Auswahl in Fülle an allen Ecken, Mitten und Enden der Linien im Fangkreis des Mauszeigers. AutoCAD zwingt sogar den Anwender, über eine Werkzeugleiste „Objektauswahl“ zu entscheiden, welche der möglichen Marker aktiv sein sollen. Punktobjekte sind gegenüber temporären Marker daher dann die effizientere und betriebssicherere Lösung, wenn sie nach der Erzeugung nicht nur einmal, sondern häufiger genutzt werden.

Für das Ziel „einfachste und effizienteste Nutzoberfläche“ sind folglich dem Ingenieur am Bildschirm Punktobjekte und Marker als sich ergänzende Lösungen zur Wahl anzubieten.

Von den Methoden, die Bauobjekte allgemein benötigen, sind für Punktobjekte die in der folgenden Liste unterstrichenen Elemente sinnvoll. Für sie sind Griffe oder Symbole zu gestalten und durch objektspezifisch intelligente Methoden zu realisieren:

Erzeugen, Identifizieren, Informieren, Einrasten, Löschen, Eigenschaften (Attribute) Ändern, Verschieben, Drehen, Kopieren, Klonen, Spiegeln, Detaillieren, Editieren,

Erweitern, Reduzieren, Verbinden (Gruppieren), Darstellen, Bezeichnen, Undo, Redo, Onlinehilfe.

Für das Punktobjekt, das selbst als sechseckiger Stern oder Nullkreis dargestellt wird, wird nur das Verschiebekreuz als Funktionsgriff vorgeschlagen:

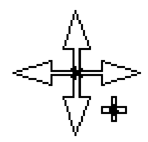


Bild 5-1: Funktionsgriff: Verschiebekreuz für „Punktobjekt“

Somit lässt sich ein Punkt direkt am Griff in der Ebene verschieben und bei gedrückter Taste <Strg> kopieren. Für die Vielzahl weiterer Befehle, die auf Punkte anwendbar sind, dient der Mauszeiger mit Spurvektor und Eigenschaftsfenster. Dieses muss neben den Spureigenschaften die in Bild 5-2 gezeigten Informationen enthalten, um einheitlich für nahezu alle Punktbefehle dienen zu können.

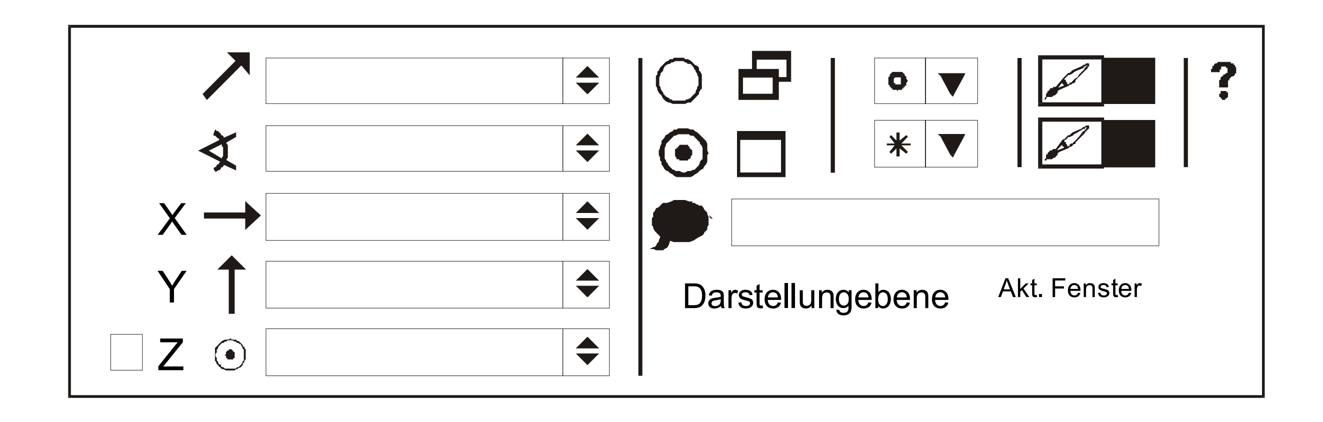


Bild 5-2: Gestaltungskonzept des Eigenschaftsfensters „Punkte“

Über die beiden Optionsfelder wird entschieden, ob ein Punktobjekt global in allen Fenstern gültig sein soll (Symbol) oder nur lokal im aktuellen Arbeitsfenster (Symbol). Dazu können jeweils über Listenfelder die Form und die Farbe für die Darstellung der Punkte auf dem Bildschirm gewählt werden. Zum Punkt ist ein Eingabefeld für Kommentare vorgesehen, symbolisiert durch . Ganz rechts ist das Fragezeichen für die Onlinehilfe des aktuellen Punktebefehls angeordnet.

Das **Erzeugen** und **Platzieren** von Punktobjekten ist im untersuchten System schon

bisher durch eine den Hauptbedarf deckende Symbolleiste selbsterklärender Symbole abgedeckt, siehe Bild 5-3. Quickinfo bei Verweilen des Mauszeigers auf einem Symbol und Onlinehilfe erläutern die Funktion einzelner Symbole mit Bildern und Schulungsbeispielen, wobei Internet-übliche Browser als Nutzoberfläche der Hilfe dienen.

Auf diesen Basisleistungen aufbauend wird folgende verbesserte Konzeption vorgeschlagen: Die einzelnen Symbole werden einheitlich so gestaltet, dass darin vor der Aktion vorhandene (vorausgesetzte) Punkte rot und durch die Funktion neu erzeugte Punkte blau dargestellt sind. Sie erscheinen hier im Schwarzweißdruck hellgrau, während rote Punkte satt dargestellt werden. Anstelle von zwei vorhandenen Punkten zur Definition einer Basislinie kann alternativ auch eine vorhandene Linie gewählt werden, wenn sie nicht am Ende der Linie geklickt wird (Linienenden wirken als Punkte). Bei Funktionen, die exakte Zahlenwerte erfordern, erscheint stets in der oberen linken Ecke des Bildschirms das Eigenschaftsfenster mit dem Eingabefeld, das die der Mausbewegung entsprechende Spurvektorlänge in Einheiten wahrer Größe anzeigt. Als metrische Einheiten des Eingabefelds sind Millimeter voreingestellt, es können jedoch auch US-Einheiten oder wertabhängige Einheiten gewählt werden. Bei wertabhängigen Einheiten, z. B. unter 1000 mm in Millimetern, darüber in Metern, ist mitlaufend eine Vergröberung des Einrastens auf Vorzugsmaße sinnvoll. Neben der mauszeiger-gesteuerten Spurvektorlänge zeigt das Eigenschaftsfenster die Koordinaten des ersten gewählten Punktes und seinen über zwei Optionsknöpfe wechselbaren Status

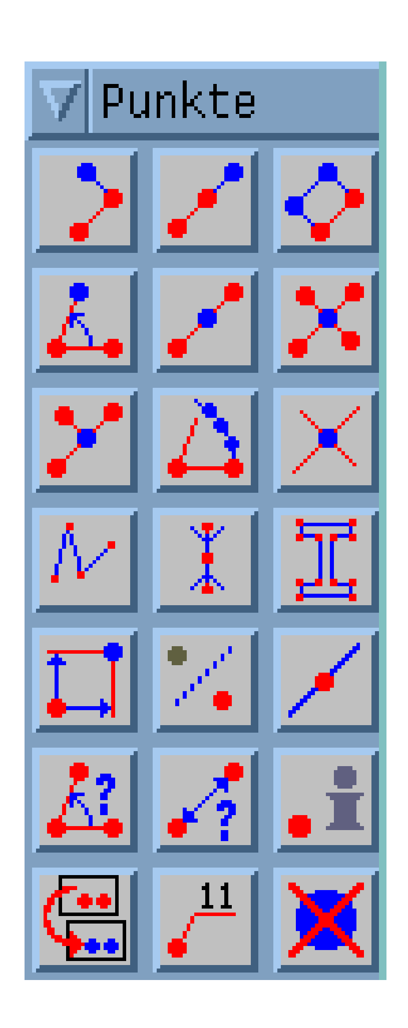


Bild 5-3: Punktsymbole

(global oder lokal fenstergebunden). Gesonderte Funktionen für „Punkte-Information“ und die Änderung der Eigenschaft „Punkt global/lokal fenstergebunden“ werden so eingespart.

Damit die Punkte-Symbolleiste nur dann Platz am Bildschirmrand belegt, wenn sie aktuell benötigt wird, ist die gesamte Leiste durch ein einziges permanentes Punkte-Symbol repräsentiert. Nach Klick auf dieses Symbol erscheint die Symbolleiste Bild 5.3 mit den zuletzt meistgenutzten Punkte-Symbolen und dem Symbol „>>“, dessen Wahl alle seltener genutzten Symbole ergänzt.

Mit einem erweiterten Ansatz kann die gesamte Punkt-Symbolleiste eingespart werden. Noch effizienter und direkter ist nämlich die Nutzoberfläche für Punktobjekte zu gestalten, wenn unterschieden wird, ob der Nutzer einen, zwei oder drei Punkte gewählt hat. Bei Wahl von zwei Punkten liegt nämlich eine die Punkte verbindende Strecke vor, zu der über Griffe direkt am Objekt ein neuer Punkt auf einer Senkrechten errichtet und zwei neue Punkte parallel in einem mausgesteuerten Abstand erzeugt werden, ein Winkel abgetragen wird oder ein Punkt in Streckenmitte entsteht.

Hat der Nutzer drei Punkte oder eine Linie und einen Punkt gewählt, wird das Lot auf die Stecke gefällt. Bei Wahl von vier Punkten oder zweier Linien ist ebenso eindeutig, dass der Schnittpunkt beider durch die Punktpaare gebildeten Strecken bzw. Linien zu erzeugen ist. Die Funktionen „Winkel messen“ und „Abstand messen“ können zusammengefasst werden, da über die Anzahl der gewählten Punkte, drei oder zwei, die Entscheidung eindeutig ist.

Bei der Konzeption der Nutzoberfläche der durch zwei Punkte gegebenen Strecken ist vorausschauend zu bedenken, dass ihre Funktionen und Hilfslinien, wo immer möglich, genau so gestaltet werden, wie bei dem in Abschnitt 5.2 entwickelten Linienobjekt und sogar den körperlichen Objekten Rasterlinie und Profilstab des Abschnitts 5-3.

Im Übrigen sind alle Punkte-Funktionen im System BOCAD-3D nochmals im vollständigen Menü „Punkte“ der Hauptmenüleiste enthalten. Dieses textbasierte, sprachabhängige Menü in BOCAD-3D wird komplett obsolet, sobald die o. g.

textfreie Symbolleiste mit Quickinfo und Onlinehilfe pro Element bzw. der erweiterte, direkte Ansatz verwirklicht ist.

Zum **Löschen** eines Punktobjekts dient bisher das Symbol ,,durchgestrichener Punkt" (Bild 5-3 unten rechts) oder der Menübefehl „Punkt löschen“. Beide Möglichkeiten sind weniger effizient und bekannt als die Taste <Entf>. Sie dient vorrangig, auch hier, zum Löschen von Objekten.

Zum **Eigenschaften ändern** wird wie bei allen Objekten durch Klick auf das Objekt mit der rechten Maustaste (Rechtsklick) das objektspezifische Eigenschaftsfenster angeboten. Eigenschaften des Punkteobjekts sind Status (global oder lokal fenstergebunden), Farbe, Form (Stern *), Punktkommentar (Punktnummer als Kommentartext voreingestellt). Das Eigenschaftsfenster für Punkte hat daher das in Bild 5-2 dargestellte Gestaltungskonzept.

Zum **Verschieben** und **Kopieren** dient direkt am Objekt der Verschiebegriff, ein Pfeilkreuz, siehe Bild 5-1. Beim Ziehen des Pfeilkreuzes mit gleichzeitig gedrückter Taste <Strg> wird kopiert, der Ursprungspunkt bleibt dann ortsfest. Beim Ziehen erscheint stets in der oberen linken Ecke des Bildschirms ein Wertefenster mit den relativen Verschiebewerten zum Ursprungspunkt als X-, Y- und Z-Wert sowie dem Abstand, jeweils in Millimetern wahrer Größe. Die Funktion ist also identisch mit dem Menübefehl „Addiere“. Das Überschreiben eines Wertes im Wertefenster führt zu entsprechender Korrektur des verschobenen bzw. kopierten Punktes.

Wenn beim Verschieben oder Kopieren die Winkelliste angeschaltet ist, werden die Winkel mit temporären Hilfslinien als Strahlenbüschel angezeigt und die als „Gummiband“ angezeigte Bewegungsrichtung des Mauszeigers beim Ziehen auf den nächstliegenden Winkel eingerastet.

Zum Kopieren eines Punktes in ein fremdes Fenster wird der Punkt aus dem aktuellen Fenster in das fremde Fenster gezogen und dort das Ziehen beendet. Der kopierte Punkt übernimmt die aktuellen Eigenschaften, d. h., wenn das Original fenstergebunden war, wird die Kopie im fremden Fenster ebenfalls an dieses Fenster gebunden. Diese Leistung ist identisch mit dem Menübefehl „Kopiere in fremdes Fenster“.

Spiegeln eines Punktmusters erfolgt wie Kopieren, jedoch wird zusätzlich durch eine Linie, zwei Punkte oder Marker eine Spiegelungsachse vorgegeben. Der Spiegel, um den gespiegelt wird, steht senkrecht zur aktuellen Arbeitsebene.

Verbinden (Gruppieren) und **Vereinzeln** von Punkten, z. B. zum Bilden von Punktmustern für Schraubenanordnungen, die als benannte Punktmuster zu speichern sind, wäre eine zwar sinnvolle Entwicklung, ist aber nach Auskunft erfahrener Konstrukteure über typisierte Teile besser zu lösen.

Undo und **Redo** fehlten früher für Punkte im System BOCAD-3D. Ersatzweise ist Löschen über ein Symbol zwar möglich, aber durch die zu bestätigende Sicherheitsabfrage und die Unüblichkeit ist diese Ersatzlösung nicht im Sinne einer einfachen Nutzoberfläche. In Verbindung mit dem Re-engineering des Systems und der Neugestaltung mit den Möglichkeiten einer objektorientierten Datenbank auf Basis der Software Objectivity wurde dieses durch historischen Ansatz bedingte Problem elegant lösbar. Hierbei ist auch zu ergänzen, dass Punktobjekte nicht nur über ein Löschsymbol und Sicherheitsabfrage löschbar sind, sondern nach Identifizierung wie bei allen Objekten üblich durch die Taste <Entf> und ggf. „Undo“ bei Irrtum.

Als Eigenschaft eines Punktobjekts muss erhalten bleiben, dass es alternativ als globaler Punkt (Datenbank-Punktobjekt) in allen bildlichen Darstellungen dauerhaft verfügbar ist, in deren Darstellungsraum es liegt, oder aber als fenstergebundener Punkt in nur einem Fenster auf dem Bildschirm, d. h. seine Lebensdauer und Erscheinung ist an dieses Fenster gebunden. Für Hilfszeichnungen hat sich diese Möglichkeit als arbeitssparend und übersichtlich bewährt.

Punktobjekte, naturgemäß ohne räumliche Ausdehnung, werden auch für fehlsichtige Nutzer gut erkennbar als sternförmiges Symbol dargestellt. Globale Punkte sind zusätzlich durch eine Punktnummer gekennzeichnet. Anhand dieser Punktnummer ist bei gleichzeitig mehreren Bildschirmfenstern mit Punkten sichtbar, ob ein Punkt in den verschiedenen Fenstern identisch ist. Die Punktnummer wird nur auf dem Bildschirm dargestellt, jedoch bei Ausgabe einer Zeichnung auf den Plotter automatisch unterdrückt. Ein identifizierter Punkt ändert zur visuellen Rückmeldung seine Darstellung vom Sternsymbol in ein gut erkennbares Kreissymbol.

Wenn der Ingenieur am Bildschirm nachträglich entscheidet, den Maßstab der Darstellung einer Zeichnung ändern zu lassen, müssen konsequenterweise auch die Orte der Darstellung von temporären Punktobjekten entsprechend neu berechnet werden.

5.2 Gerichtete und zweidimensionale Objekte

Selbst in CAD-Systemen, die auf einem rechnerinternen 3D-Volumenmodell beruhen, sind in nicht zu vernachlässigender Häufigkeit 2D-Grafiken interaktiv zu ergänzen. Untersuchungen im Rahmen dieser Arbeit bei Anwendern führten bei einigen Firmen sogar zu einem überraschenden Ergebnis. Auftragsklärung mit Klärungsblättern als Montage von vorgefertigten „Klebefolien“ ist angeblich wirtschaftlicher und wesentlich schneller, als Auftragsklärung und entsprechend häufige Änderung des 3D-Modells. Erst nach Klärung mit 2D-Bildern lohnt angeblich die 3D-Konstruktion. Betrachtet man die Detailanforderungen des für diese Leistung typischen Bildes 5-4, ist dieser Effekt nicht völlig auszuschließen.

Auch dreidimensional arbeitende 3D-Systeme der oberen Leistungsklasse müssen daher ein ebenso leistungsfähiges Angebot an 2D-Leistungen enthalten.

Einfachheit und Effizienz der Nutzoberfläche durch intelligentes Verhalten der 2D-Objekte sind hier ebenso wichtig wie bei 3D-Objekten, da gerade 2D-Objekte in hohem Maße interaktiv vom Ingenieur am Bildschirm in nicht automatisierbaren Schritten erzeugt und modifiziert werden.

In den folgenden Abschnitten werden die einzelnen 2D-Objekte erläutert. Gemeinsamkeiten und Besonderheiten werden vorab zusammengefasst.

Linien sind ähnlich zu behandeln wie zweidimensionale Objekte, z. B. Texte oder „Klebefolien“. Sie sind zwar eindimensional, aber gerichtet. In CAD-Systemen mit 3D-Volumenmodell, in dem die Kanten von Körpern automatisch dargestellt werden, ist sorgfältig zu unterscheiden zwischen Linienobjekten und den als Linien dargestellten Kanten von Körperobjekten. Ein Linienobjekt ist eigenständig, also nicht nur Element eines übergeordneten Objekts.

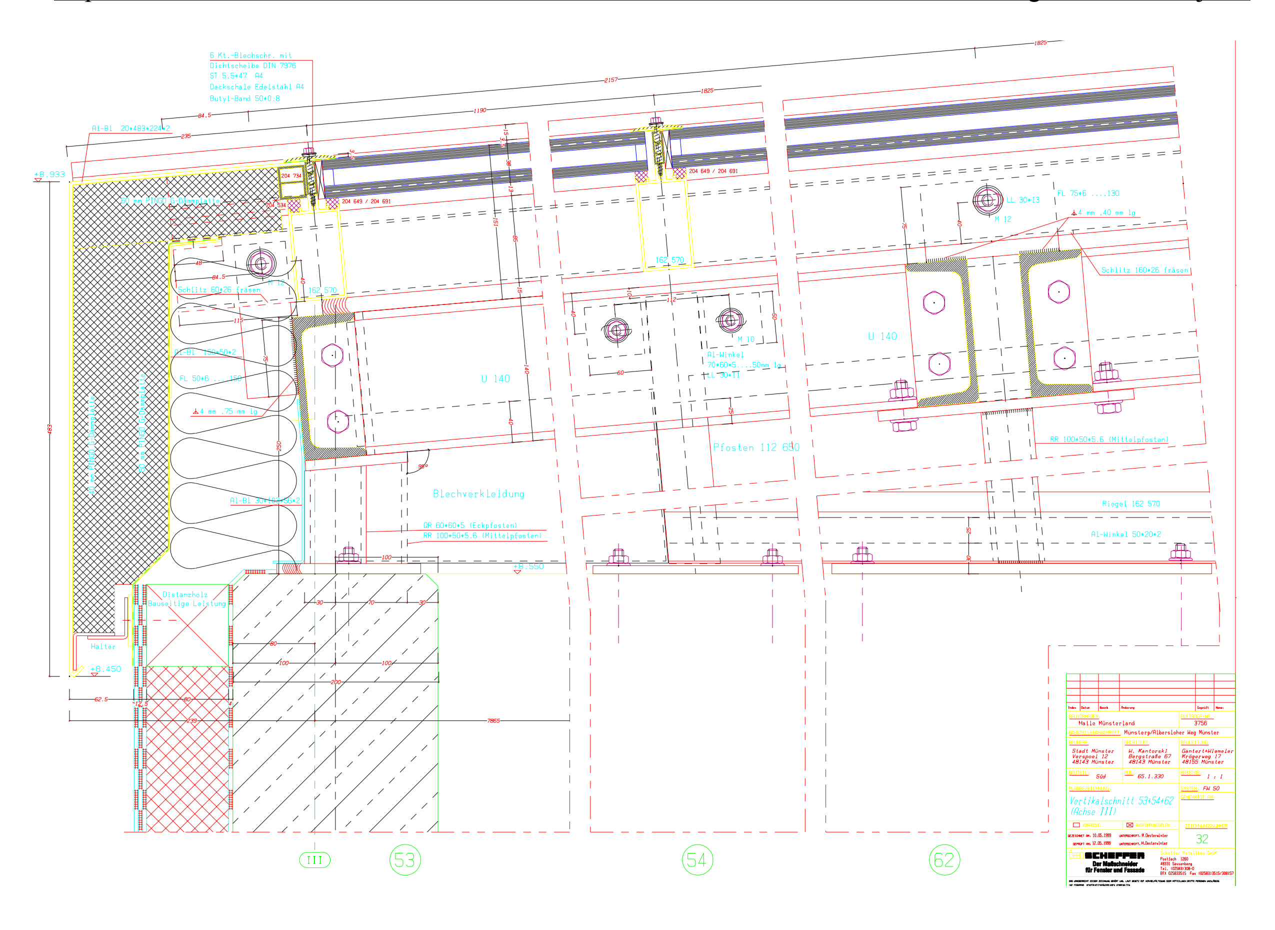


Bild 5-4: Klärungsblatt mit hohen Anteilen an 2D-Grafik

Gerichtete und zweidimensionale Objekte, also 2D-Grafik, besitzen Gemeinsamkeiten, die sie von anderen Objektklassen, etwa 3D-Körpern, unterscheiden. Dazu zählen die organisatorischen, strukturellen Aktionen eines Gruppeneditors mit „Gruppieren“ zu einem Segment und „Gruppierung auflösen“ in einzelne Primitiva, ,,Element zur Gruppe hinzufügen“ und entsprechend „Element herauslösen“. Wie in Abschnitt 5.2.6 näher erläutert, sind häufig externe Metafile-Dateien, z. B. im DXF-Format, mit 2D-Grafik zu importieren und dabei in eine Gruppe umzuwandeln sowie die umgekehrte Aktion, Gruppen als Metafile zu exportieren.

Kopieren und Spiegeln von 2D-Grafik müssen die Struktur bewahren. Eine Liste von Primitiva, die nicht zu einem Segment zusammengefasst waren, sollen nach Spiegeln oder Kopieren weiterhin einzelne Primitiva bleiben. Das gleiche gilt für Segmente und zu einer Gruppe zusammengefasste Segmente.

Gerade bei umfangreicheren Ergänzungen einer Zeichnung mit 2D-Grafik kann die

geplante Zeichnungsgröße wider Erwarten überschritten werden. Dann muss die Zeichnung ohne Verlust der interaktiven Arbeiten für die 2D-Ergänzungen mit einem neuen Maßstab dargestellt werden, gewählte Schriftgrößen jedoch sind beizubehalten. Dazu muss jede 2D-Grafik mitskaliert und der Ursprung mittransformiert werden, auch temporäre 2D-Grafik, die nicht in der Datenbank gespeichert ist.

Besonders für 2D-Aktionen, die naturgemäß alle als interaktive Ergänzungen handwerklich aufwendig sind, muss Undo und Redo möglich sein.

Die auch für 2D-Grafikobjekte sinnvolle Aktion „Dehnen/Stauchen“ erfordert analog zur 3D-Aktion erhebliche Objektintelligenz. Da für Klärungsblätter mit 2D-Darstellung von Kantblechen gedacht, wird vorgeschlagen, tatsächlich 3D-Kantbleche als schmale Streifen zu verlegen und hierfür das Dehnen und Stauchen effizienter zu gestalten. Auch der sinnvolle Anwendungsfall ist bekannt, dass die Bauhöhe eines als „Klebefolie“ vorgegebenen Fußbodenaufbaus nachträglich vom Architekten geändert wird, also die komplette Fußbodengrafik gedehnt werden muss.

5.2.1 Linie, Polygon und geschlossenes Polygon, Kreissegment

Linie, Polygon und geschlossenes Polygon haben so viele Gemeinsamkeiten, dass sich zur Vereinfachung und Vereinheitlichung eine gemeinsame Nutzoberfläche anbietet. Systematisch und konsequent ist darauf aufbauend auch die Ähnlichkeit von Profilstäben und Linien durch gemeinsame, einheitliche Gestaltung der Funktionsgriffe zu nutzen. Das gleiche gilt für Polygone und Blechkonturen bzw. Stirnansichten von Blechprofilen mit Parallelkantungen. Dadurch erhöht sich der Wiedererkennungswert und der Lernaufwand verringert sich beträchtlich. Zusätzlich ist der Ansatz interessant, eine Linie bzw. Stabachse durch einen Exzentrizitäts-Griff als Durchmesser-Symbol „ø“ in Achsenmitte zu einem Kreisbogen zu biegen.

Eine Linie benötigt Griffe oder Symbole für folgende Funktionen, die durch objektspezifisch intelligente Methoden zu realisieren sind:

Erzeugen, Platzieren, Löschen, Eigenschaften Ändern, Dehnen/Stauchen, Zerschneiden in zwei Objekte, Verschieben, Kopieren, Spiegeln, Biegen, Verbinden

(Gruppieren) und Vereinzeln, Importieren und Exportieren, Undo und Redo.

In Corel Draw sind für das Erzeugen von Objektgruppen, darunter Freihandlinien, Linien und Polygone, objektspezifische Eigenschaftsleisten als Kommandozeile verwirklicht, die die Prinzipien Textfreiheit und Verständlichkeit ohne Lernaufwand annähernd einhalten, Bild 5-5.

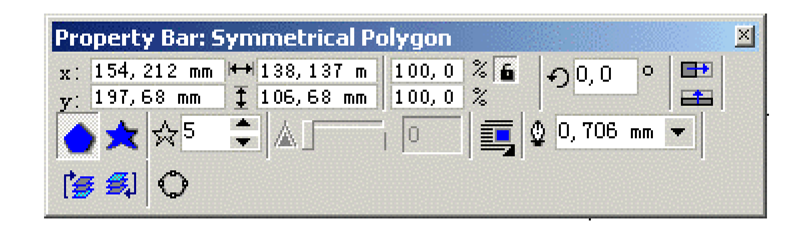


Bild 5-5: Eigenschaftsleiste für Linien (Quelle: Corel Draw)

Demgegenüber ist bei 3D-Systemen, die nur ergänzend auf 2D-Objekte, wie Linien, zurückgreifen müssen, das Eigenschaftsfenster „Polygon“, übersichtlicher zu gestalten, Bild 5-6.

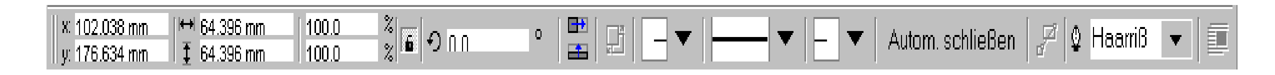


Bild 5-6: Eigenschaftsfenster „Symmetrisch Polygon“ (Quelle: Corel Draw)

Corel Draw bietet für Linienobjekte Anfasser in den Eckpunkten und Kantenmitten des zur Arbeitsebene achsparallelen, umschreibenden Rechtecks sowie Funktionsgriffe direkt auf der Linie an den Linienenden und in der Linienmitte, Bild 5-7.

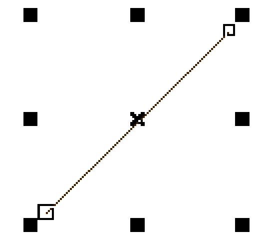


Bild 5-7: Funktionsgriffe (Quelle Corel Draw)

Die Anfasser des umschreibenden Rechtecks zeigen zum Einen an, dass das Objekt markiert ist. Zum Anderen dienen diese Anfasser zur Vergrößerung bzw. Verkleinerung des Objekts. In einem CAD-System wie BOCAD-3D wird das Markieren eines Objekts durch Aufleuchten des Objekts selbst wesentlich

anschaulicher und ohne verwirrende Überlappungen bei mehreren markierten Objekten angezeigt. Da zudem alle Darstellungen maßstäblich sind, ist das Vergrößern oder Verzerren eines Linienobjekts über diese Anfasser nicht brauchbar. Anfasser entfallen deshalb in vorteilhafter Weise.

Folgende Funktionsgriffe werden daher für das Linienobjekt vorgeschlagen:

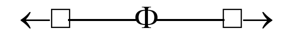


Bild 5-8: Funktionsgriffe am Linienobjekt

Eine Besonderheit ist die „Querlage von Linien für Absperrfolien“: Absperrfolien, die mittig auf Teilekanten liegen, sind bei größerer Kanten-Strichstärke praktisch nicht mehr erkennbar. Sie sind daher mit der Spezifikation „rechts“ bzw. „links“ von Teilenkanten der 3D-Körper oder mit zusätzlicher Querverschiebung zu verlegen.

Die Ähnlichkeit von Blechkonturen und geschlossenen Polygonen kann konsequent durch einheitliche Gestaltung der Funktionsgriffe zur Erhöhung des Wiedererkennungswerts und zur Verringerung des Lernaufwands genutzt werden. Beide benötigen Griffe oder Symbole für gleichartige Funktionen, die durch objektspezifisch intelligente Methoden zu realisieren sind:

Erzeugen, Platzieren, Löschen, Eigenschaften Ändern, Kontur Ändern inkl. Dehnen/Stauchen, Zerschneiden in zwei Objekte, Verschieben, Kopieren, Spiegeln, Biegen, Verbinden und Vereinzeln.

5.2.2 Kreis, Ellipse, Parabel

Bild 5-9 zeigt die allgemeine Form des Objekts „Kreis“ mit seinen Griffen. Das Erzeugen eines Kreises erfolgt durch Ziehen des Kreissymbols aus den 2D-Werkzeugen und Wählen des Zielorts für den Mittelpunkt. Über den Radiusgriff wird der voreingestellte Kreisradius auf das gewünschte Maß gezogen, das im Statusfeld angezeigt und dort ggf. noch quantitativ exakt eingestellt wird.

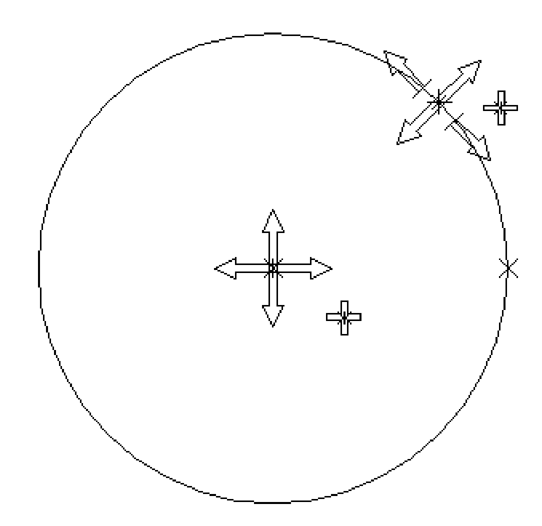


Bild 5-9: Griffe für Objekt Kreis

Gibt man im Statusfeld eine Radiusliste an, werden in einer Aktion gleich mehrere konzentrische Kreise erzeugt. Alternativ sind konzentrische Kreise über den Radiusgriff durch Kopie bei gedrückter Taste <Strg> zu erzeugen.

Ziehen des quer zum Radiusgriff angebotenen linken bzw. rechten Teilers auf den gewünschten Start- und Endpunkt ändert den Kreis zum Kreissegment. Die Öffnung liegt stets zwischen den Teilertrennstrichen.

Ziehen des angebotenen Ellipsengriffs macht den Kreis zur Ellipse. Taste <Entf> löscht den gewählten Kreis.

Kreissegmente werden über die Verbindungslinie der Segmentendpunkte und den Durchmessergriff der Verbindungslinie erzeugt und ggf. geändert. In Corel Draw ist dazu folgende Lösung gewählt, Bild 5-10:

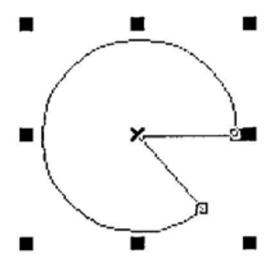


Bild 5-10: Kreisbogen durch Griffe am Kreis herstellen (Quelle Corel Draw)

Zu diesen Anfassern und Griffen an Kreisen wird eine entsprechende Eigenschaftsleiste angeboten, Bild 5.11:



Bild 5-11: Eigenschaftsleiste Kreisbogen (Quelle Corel Draw)

Die Symbole in dieser Eigenschaftsleiste bewirken, dass das Objekt Ellipse bzw. Kreis (als Sonderfall der Ellipse) als Kreissegment oder Kreisbogen dargestellt wird und welche Winkel in Grad ° zur lokalen X-Achse die offenen Flanken bilden. Über die Schaltflächen ▲ und ▼ kann der jeweilige Winkel in Inkrementen geändert werden.

5.2.3 Textblock

Bild 5-12 zeigt die allgemeine Form des Objekts „Textblock“ mit Griff. Textblöcke können ein- oder mehrzeilig sein. Textblöcke ohne Anriss werden im Bauwesen für Bemerkungen und Hinweise wie „bauübliche Toleranzen beachten“ verwendet, also wie Stempel. Sie beziehen sich entweder allgemein auf alle in der Zeichnung dargestellten Objekte oder sind durch ihre Platzierung eindeutig erkennbar einem bestimmten Objekt zuzuordnen.

Zu den Eigenschaften eines Textblocks zählen Schrifthöhe, Schriftart, Umrandung des Blocks, Opaque-Eigenschaft und der Ankerpunkt, mit dem der Textblock platziert wurde.

Die Identifizierung eines Textblocks erfolgt durch Rechtsklick auf die linke untere Ecke des Textblocks. Zur Bestätigung erhält der Textblock in seinem individuellen Ankerpunkt einen Verschiebegriff, der Text erscheint im Editierfenster, die Eigenschaften im Eigenschaftsfenster.

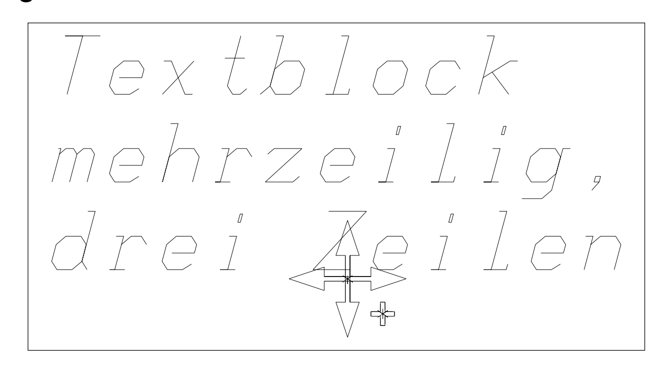


Bild 5-12: Anfasser für Textblock

Folgende direkte Aktionen am Objekt ,,Textblock“ werden dadurch möglich: Löschen durch <Entf>, Verschieben, Kopieren, Editieren des Textes und Eigenschaften des Textblocks ändern.

Der Verschiebegriff bewegt den Textblock an einen neuen Ort. Handelt es sich um einen Textblock mit Anrisslinie, wird der Knickpunkt der Anrisslinie „intelligent“ mitgezogen.

Die gleiche Aktion, jedoch gezogen bei gleichzeitig gedrückter Taste <Strg>, kopiert den Textblock einschließlich aller aktuellen Eigenschaften an den Zielort.

Der Text selbst ist im Editierfenster änderbar. Seine Eigenschaften sind im Eigenschaftsfenster zu ändern, wobei jede einzelne Änderung sofort mitlaufend in der Zeichnung angezeigt wird.

Mehrzeilige Textblöcke müssen im Ganzen als Gruppe kopierbar und editierbar sein, z. B. wie mit dem bekannten Programm Notepad des Betriebssystems Windows.

Beim Kopieren von 2D-Grafik in ein fremdes Fenster müssen intelligente Funktionen die Arbeit des Ingenieurs am Bildschirm vertretbar halten. Dies bedeutet, dass die kopierte Grafik den Zielmaßstab und die Zielschrifthöhe automatisch berücksichtigen muss. 2D-Grafik, die von einer Ursprungszeichnung mit Ursprungsmaßstab in eine Zielzeichnung mit anderem Zielmaßstab kopiert wird, muss sich also automatisch entsprechend dem Zielmaßstab skalieren. Ausgenommen sind Texte, denn diese sollen sinnvollerweise ihre Schrifthöhe beibehalten. Die Skalierung bzw. Nichtskalierung ist somit objektabhängig.

5.2.4 Textblock mit Anriss

Bild 5-13 zeigt die allgemeine Form eines „Textblocks mit Anriss“, der am Ankerpunkt mit seiner Anrisslinie auf ein anderes Objekt zeigt. Textblöcke mit Anriss werden im Bauwesen für Hinweise wie ,,bauseits anpassen“ verwendet, die sich auf ein bestimmtes Objekt oder mehrere bestimmte Objekte beziehen. Falls sich der Textblock auf mehrere Objekte bezieht, zeigen entsprechend viele Anrisslinien dorthin (Bild 5-14), sofern die Objekte benachbart, d. h. in sinnvoller Nähe sind.

Der Textblock mit Anriss besteht aus einem Textblock und dem Anriss. Der Textblock selbst ist wie der einfache Textblock, Abschnitt 5.2.3, zu identifizieren und zu bearbeiten. Der Textblock ist also nichts anderes als ein Objekt Textblock im übergeordneten Objekt Textblock mit Anriss.

Die Identifizierung eines Textblocks mit Anriss erfolgt durch Rechtsklick auf die Anrisslinie, denn dadurch unterscheidet er sich von einem Textblock ohne Anriss. Zur Bestätigung erhält die Anrisslinie in ihrem Knickpunkt einen Verschiebegriff und im Ankerpunkt einen Ankergriff. Wenn nicht die Anrisslinie, sondern der Textblock selbst geändert werden soll, ist der Textblock zu identifizieren, wie in Abschnitt 5.2.3 definiert.

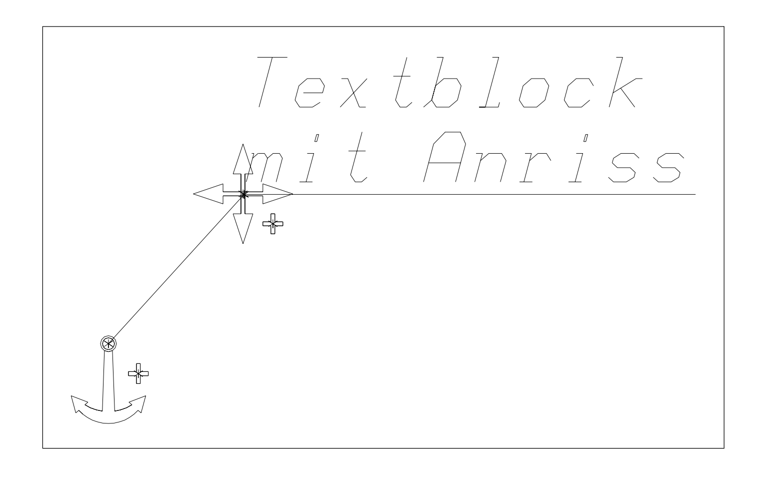


Bild 5-13: Anfasser für Textblock mit Anriss

Vier direkte Aktionen werden nach der Identifizierung möglich (neben dem Löschen durch <Entf>). Der Verschiebegriff bewegt den Knickpunkt der Anrisslinie bei festem Anker an einen neuen Ort. Der horizontale Teil der Anrisslinie bleibt dabei horizontal, die Linie zum Anker bewegt sich gummibandartig mit. Wird so verschoben, dass die Distanz vom Ankerpunkt zum Ende des horizontalen Linienteils kürzer ist als zu ihrem Anfang, dann wechselt die Ankerlinie die Seite entsprechend, also „intelligent“.

Die gleiche Aktion, jedoch gezogen bei gleichzeitig gedrückter Taste <Strg>, kopiert den Textblock mit Anriss einschließlich aller Eigenschaften in seiner aktuellen Form an einen neuen Zielort.

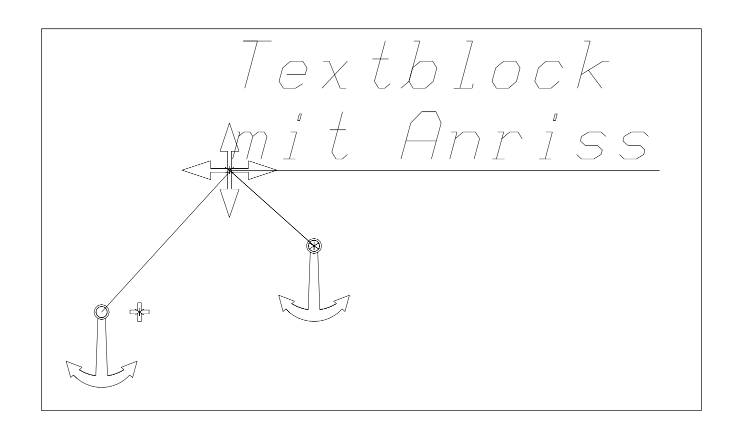


Bild 5-14: Mehrfachanriss durch <Strg>

Ziehen des Ankergriffs an einen neuen Zielort zieht die Ankerlinie gummibandartig mit, alle übrigen Elemente bleiben unverändert.

Die gleiche Aktion, jedoch gezogen bei gleichzeitig gedrückter Taste <Strg>, erzeugt eine zusätzliche Ankerlinie mit Anker, so dass zwei Ankerlinien vom Knickpunkt ausgehen, Bild 5-14. Ebenso sind weitere Ankerlinien zum Textblock zu erzeugen.

Kopieren und Spiegeln eines Textblocks mit Anriss als Ganzes erfolgen intelligent über den Textblock wie unter Abschnitt 5.2.3 beschrieben, da der Textblock den Anriss in unveränderter Form beim Kopieren mitzieht. Beim Spiegeln wird in gespiegelter Form mitgezogen.

Gemeinsame Eigenschaft aller Textblöcke mit und ohne Anriss ist das Verhalten als opaque Flächen, d. h. sie überdecken mit ihrer Fläche die Darstellung anderer Objekte im Überlappungsbereich. Wird der Textblock nachträglich interaktiv verschoben, erscheint das überdeckte Objekt wieder.

5.2.5 Textobjekte fluchtend anordnen

Textobjekte werden im CAD-System für Bemerkungen und Hinweise verwendet, dienen also dem sogenannten „Papier-Bereich". Während der Konstruktion muss der Anwender vor dem Ausdruck der endgültigen Unterlagen solche Texte häufig ändern, um die Angaben zu präzisieren bzw. „die Papiere sauber zu machen". Im untersuchten CAD System können durch die Dialogbox „Text editieren" alle Attribute

von Texten, beispielsweise Textinhalt, Schrifthöhe und Schriftneigung, editiert werden. Es werden aber nur diejenigen Textattribute editiert, deren Spezifikation durch ein eigenes Kontrollfeld auch eingeschaltet ist.

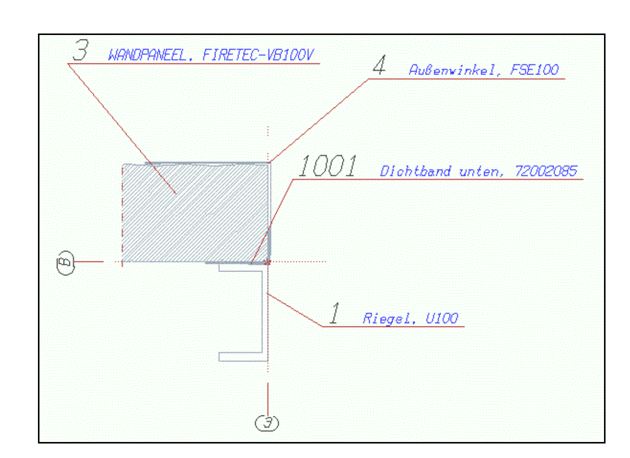
Ein Textobjekt kann wie ein Grafikelement verschoben werden. Dies genügt aber nicht in der Baupraxis. Nach Erfahrungen aus dem Alltagseinsatz muss das Verschieben von Texten „intelligenter“ durchgeführt werden, nämlich fluchtend, um eine ästhetisch befriedigende Anordnung zu erreichen.

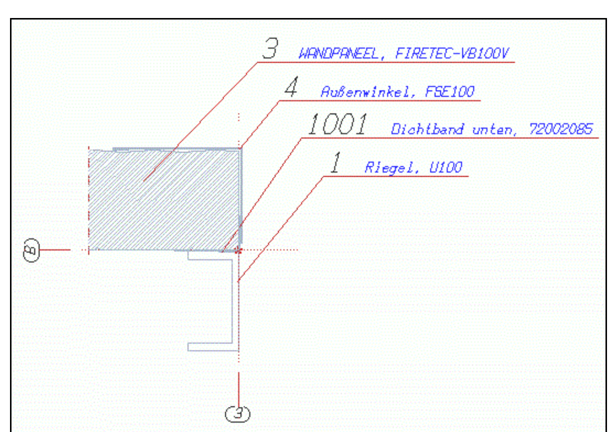
- Mehrere Textobjekte können gleichzeitig an einen neuen Ort verschoben und dabei fluchtend zu einer Linie angeordnet werden.
- Das Verschieben von Textobjekten sollte dadurch bewerkstelligt werden, dass jeweils der Knickpunkt der Anrisslinie eines Textobjektes mit Anriss auf festem Anker an einen neuen Ort platziert wird. Die ausgewählten Textobjekte werden automatisch gemäß der Pickreihenfolge neu angeordnet.
- Die Linie, zu der die Textobjekte fluchtend angeordnet werden, kann durch Eingabe des Winkels oder durch Picken zweier Punkte definiert werden.
- Diese Aktion gilt nur für die Typen „Text“, „Textblock“ und „Text mit Anriss“.
- Der Text kann rechts oder links zur Platzierungslinie platziert werden, wozu ein Punkt zur Bestimmung der Seite gepickt wird.

Die Identifizierung der Textobjekte für die fluchtende Anordnung erfolgt durch Picken auf die Objekte. Durch Picken eines Platzierungspunktes und Eingabe eines Winkels (oder Picken zweier Punkte für die Festlegung der Richtung) werden die Textobjekte an ihren neuen Ort verschoben und neu fluchtend angeordnet, siehe Bild 5-15. Das Programm kann dabei „intelligent“ erkennen, ob die ausgewählten Elemente tatsächlich Textobjekte sind. Andere Objekte werden ausgeschlossen und nicht behandelt.

Wird so verschoben, dass die Distanz vom Ankerpunkt zum Ende des horizontalen Linienteils kürzer ist als zu ihrem Anfang, dann wechselt die Ankerlinie entsprechend die Seite, verhält sich also „intelligent“. Bild 5-15 zeigt die Textobjekte vor und nach

dem Verschieben.





Vorher

Nachher
(mit der Ausrichtung 120 Grad)

Bild 5-15: Fluchtende Anordnung von Textobjekten

5.2.6 2D-Grafik importieren, exportieren, verändern

Architekten verlagern die Konstruktion architektonischer Details mehr und mehr an fachkompetente Konstruktionsbüros und Bauunternehmen. Für diese Detailzeichnungen, Klärungsblätter genannt, wird üblicherweise kein Honorar gezahlt. Deshalb muss der Aufwand für diese Klärungsblätter durch intelligente Objekte mit direkter Eingabe über Funktionsgriffe vertretbar gehalten werden. Es entstehen so zusätzliche Anforderungen an CAD-Systeme des Stahl-, Holz- und Glasbaus, insbesondere an 2D-Leistungen. Besonders im Glasbau fordern nämlich Architekten von den Büros der Bauunternehmen und Metallbauer Detail-zeichnungen, die Zukaufteile nicht wie in technischen Zeichnungen üblich als genormte Symbole, sondern naturgetreu bis hin zur bildlichen Gewindedarstellung enthalten. Eine dreidimensionale Erzeugung von Volumenmodellen derartiger Objekte würde den programmtechnischen Aufwand, den Speicherbedarf und die Antwortzeiten von CAD-Systemen der Hochleistungsklasse unvertretbar in die Höhe treiben. Daher wird das Problem durch „Klebefolien“ gelöst, die als Grafikdateien im DXF-Format von den Herstellern der Zukaufteile auf CD-ROM zur Verfügung gestellt

werden. Zwei typische Beispiele der 2D-Grafik solcher Zukaufteile einschließlich eingebetteter Werbung zeigen die Bilder 5-16 und 5-17.

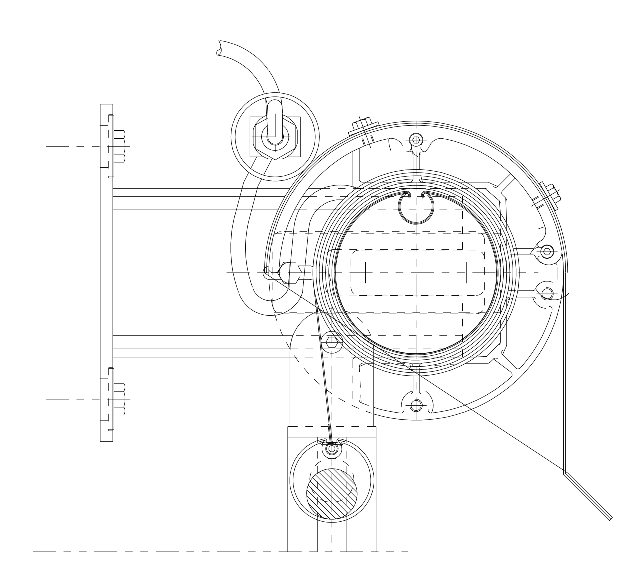


Bild 5-16: 2D-Grafik einer Zukauf-Baugruppe

Naturgetreue Bilder mit 2D-Hauptansichten von selbstschneidenden Schrauben, Dübeln etc. liegen üblicherweise als AutoCAD-Ergänzungen im DXF-Format oder zukünftig DWG-Format vor. Die industrieüblichen Grafikformate DXF und DWG enthalten leider intern keine Angabe zur verwendeten Maßeinheit und zum verwendeten Darstellungsmaßstab. Selbst der zweckmäßige Referenzpunkt zum Platzieren eines Bildes, etwa der Schnittpunkt der Schraubenmittellinie mit der Auflagerfläche, fehlt in aller Regel. In Bild 5-17 verdeckt sogar das werbende Firmenemblem, ein Fischsymbol der Firma Fischer, den als Referenzpunkt geeigneten Schnittpunkt.

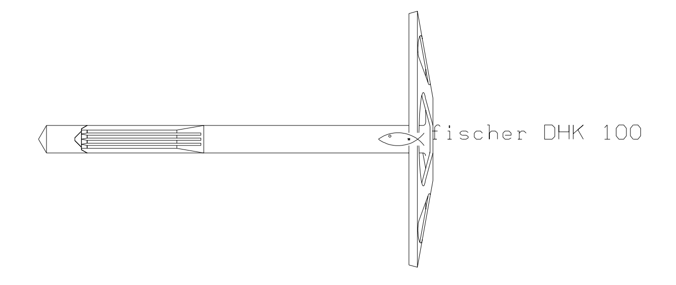


Bild 5-17: 2D-Grafik eines Zulieferteils

Für den Metall- und Stahlbau stellen viele Zulieferfirmen digitale Produktbibliotheken wie Profil-, Dübel- und Schraubenbibliotheken zur Verfügung. Dabei werden alle

gängigen 2D-Formate unterstützt, z. B. AutoCAD, Solid Edge, TurboCAD, MegaCAD sowie alle CAD-Systeme, die DXF und DWG unterstützen. Manche Zulieferfirmen bieten Programme an, die in viele im Praxiseinsatz ausgereifte CAD-Systeme integrierbar sind. Der Anwender kann damit direkt die bauspezifischen 2D-Objekte aus der Bibliothek auswählen und in die aktuelle Zeichnung einfügen. Die für Dübel bekannte Firma Fischer stellt z. B. ein solches Programm zur Verfügung (Bild 5-18). Wenn der Benutzer Dübeltyp, Nutzlänge und Maßstab ausgewählt hat, erscheint das Objekt automatisch mit Vorschau auf der Zeichnung des aktuellen laufenden CAD-Systems. Danach ist die Leistung für das Einfügen des Objekts (in AutoCAD „insert block") aufzurufen. Sobald der Benutzer Platzierungspunkt, Rotationswinkel etc. festgelegt hat, wird das Objekt als 2D-Grafikelement in die Zeichnung des Zielsystems integriert. Hier handelt es sich also um einen typischen Anwendungsfall der in Abschnitt 4.2.1 beschriebenen Mehr Fenster-Technik.

Da das hier untersuchte CAD-System eine eigenständige Grafikschnittstelle besitzt, kann die oben genannte Fischer-Dübeldatenbank nicht direkt über MDI integriert werden.

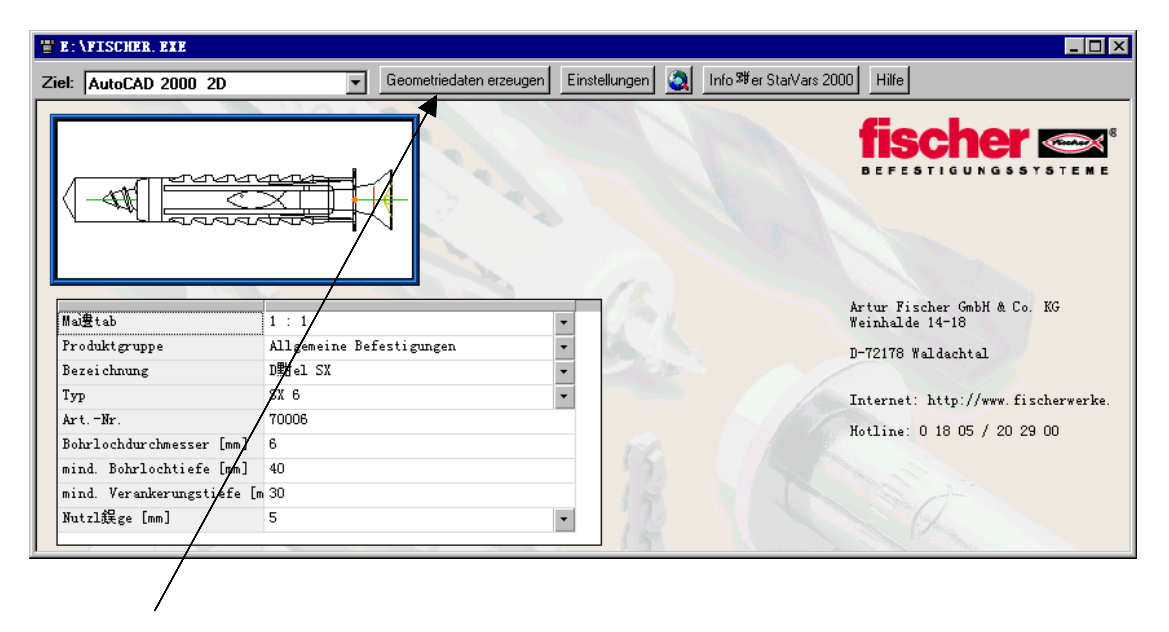
Es ist daher notwendig und sinnvoll, eine vom Zulieferer zur Verfügung gestellte CD-ROM mit 2D-Grafik einmal vorab für die spätere häufige Verwendung in konkreten Aufträgen aufzubereiten, damit eine einheitliche Ursprungseinheit, ein einheitlicher Ursprungsmaßstab und pro Bild mindestens ein geeigneter Referenzpunkt genutzt werden kann. Für diese einmalig vorzunehmende Aufbereitung, die bei mehreren Anwenderfirmen in sinnvoller Weise zentral durch den CAD-Anbieter erfolgt, ist angesichts der Masse der stark repetitiven Aufbereitungsarbeit ein passendes Hilfsprogramm bereitzustellen. Für Anwender, die die Aufbereitung z. B. aus Termingründen sofort selbst vornehmen, muss dieses Hilfsprogramm ebenfalls so leistungsfähig und intelligent entwickelt werden, dass die Bedienung einfach und effizient ist. Das direkte Hilfsprogramm zum Importieren von 2D-Grafik unter Berücksichtigung des Arbeitspunktes wurde daher folgendermaßen gestaltet: Zulieferer von Befestigungsmitteln liefern naturgetreue Bilder mit 2D-Hauptansichten von selbstschneidenden Schrauben, Dübeln etc. auf CD-Rom im Format DXF oder zukünftig DWG im Maßstab 1:1, der Konstrukteur kann ein 2D-Grafikobjekt aus der Bibliothek auswählen und in die aktuelle Zeichnung platzieren.

Die folgenden Akzeptanzkriterien wurden bei der Entwicklung berücksichtigt:

- Die 2D-Grafik soll als Vorschau gezeigt werden.
- Der Pfadname des Ordners der Vorschau soll möglichst wie im bekannten Windows-Explorer wählbar sein.
- Nach Wahl des Arbeitspunkts im gewünschten Bild der Vorschau soll dieses Bild durch Klick auf einen Zielpunkt in der Ziel-Zeichnung dort maßstabsgerecht als ein Segment erscheinen.
- Der Arbeitspunkt soll dann auf dem Zielpunkt liegen. Das Bild ist um den Arbeitspunkt drehbar.

Bevor die 2D-Grafik im Zielsystem direkt platziert werden kann, muss sie in die interne Darstellung des Zielsystems gewandelt werden. Da Bibliotheken viele Bilder enthalten, sollen zunächst beliebige Mengen von 2D-DXF-Dateien in entsprechende Dateien des Zielsystems gewandelt werden können (Bild 5-19). Im Textfeld *Kommando* kann der Benutzer den Skalierungsfaktor und die Einheit der Grafik eingeben.

DXF ist ein Industrie-Standard-Dateityp für 2D-Grafik. Die Dateien bestehen aus Abschnitten, und jeder Abschnitt enthält mehrere Gruppen, Tabelle 5-1 zeigt die Strukturierung der DXF-Datei. Bei der Wandlung in den Ziel-Dateityp wird jedes Entity zum Primitivum gewandelt, und jeder Block wird zur Gruppe gewandelt.



Durch Wählen wird 2D-Grafik erzeugt und als ein Block in AutoCAD platziert. Der Befehl „insert block“ wird automatisch aufgerufen.

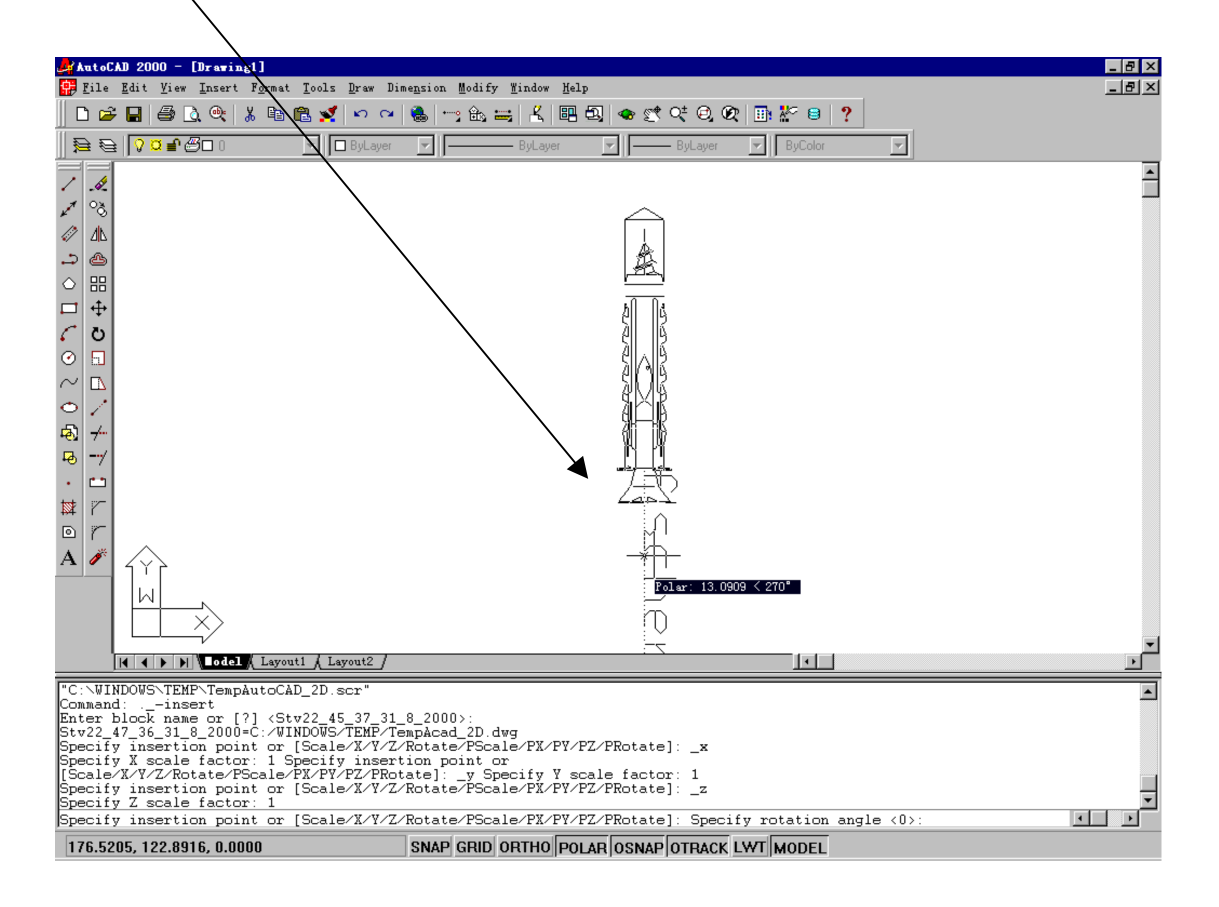


Bild 5-18: Fischer-CAD-Dübeldatenbank integriert in AutoCAD

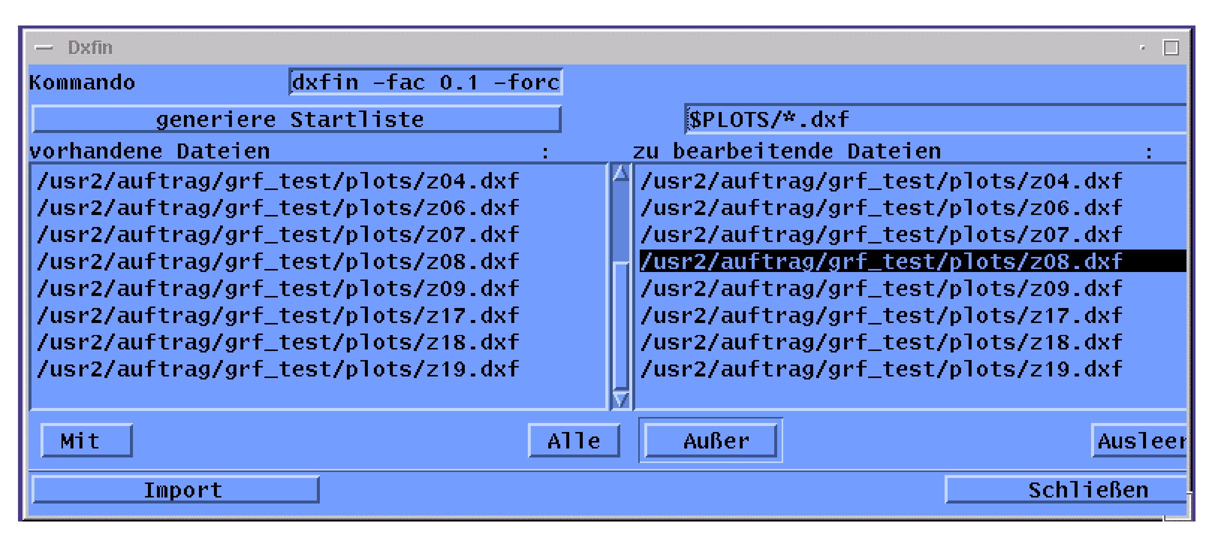


Bild 5-19: Import von 2D-DXF-Dateien

Abschnitt	*Gruppe Kode*	*Gruppe Wert*		*Abschnitt*	*Gruppe Kode*	*Gruppe Wert*	
Titel	0	SECTION	Allgemeine Information (Version, System - Variante...)	Klasse	0	SECTION	Information über Klassen
	2	HEADER			2	CLASSES	
	...	...			...	...	
	0	ENDSEC			0	ENDSEC	
Tabelle	0	SECTION	APPID, BLOCK_RECORD, DIMSTYLE, LAYER, ...	Block	0	SECTION	Information über alle Defintionen des Blocks
	2	TABLES			2	BLOCKS	
	...	...			...	...	
	0	ENDSEC			0	ENDSEC	
Entity	0	SECTION	POINT, LINE, ARC, CIRCLE, ...	Objekt	0	SECTION	alle übrige Daten, die nicht grafische Entities sind
	2	ENTITIES			2	OBJEKTS	
	...	...			...	...	
	0	ENDSEC			0	ENDSEC	
Signatur	0	EOF	Ende der Datei				

Tabelle 5-1: Strukturierung von DXF-Dateien

Das hier untersuchte CAD-System BOCAD-3D besitzt eine eigene Grafikschnittstelle, das sogenannte bmf_ Format. Aus diesem Grund kann z. B. die Dübeldatenbank der oben genannten Firma Fischer nicht direkt in das System integriert werden. Im Vergleich zu »Block« und »Entity« der DXF Datei besteht diese Grafikschnittstelle aus »Gruppe«, »Segment« und »Primitivum«. Tabelle 5.2 zeigt die Datenstruktur von »Gruppe«, »Segment« und »Primitivum« in der bmf_ Datei. Bei der Wandlung in den BOCAD-Dateityp wird jedes »Entity« zum Primitivum gewandelt, und jeder Block wird zum Segment gewandelt.

Objekt	***Datenstruktur***	***Methode***
Gruppe	int nr; /* Gruppenummer int typ; /* Gruppentyp double ruetra[12]; /* 3D-Rücktransformation float xmin,ymin; /* orthogonale Extremwerte (min) float xmax,ymax;/* orthogonale Extremwerte (max) float xminl,yminl; /* orthogonale Extremwerte (min) lokal float xmaxl,ymaxl;/* orthogonale Extremwerte (max) lokal float xtra,ytra; /* Translationen float xsca,ysca; /* Skalierungen float win; /* Rotation float trf[6]; /* akkumulierte Transformationen int date; /* Datum des letzten Update short zu; /* Gruppe, zu der Gruppe gehört short l,r,u; /* linker und rechter Nachbar im Baum short del; /* Löschkennung short reo; /* Kennung, ob Gruppe wiedereröffnet short attr; /* Vorrangattribut unsigned short seg;/* Anzahl der Segmente in dieser Gruppe unsigned short aseg;/* Zeiger auf erstes Segment der Gruppe unsigned short eseg;/* Zeiger auf letztes Segment der Gruppe unsigned short grprest[2]; /* Pad für Alignments /* bei root == log. Sammelblattnr.	*void own_mpur_ginf (double elem[], int *nwas, int *iwas, int *nr, int *typ);* *void own_mpur_ginfi_(double elem[], int *nwas, int *iwas, int *igrp);* *void own_mpur_gind_(double elem[],int *ifile, int *nr, int *typ);* *void own_mpur_gbkz_(double elem[],int *ifile, int *igrp);*
Segment	int nr; /* Segmentnummer int typ; /* Segmentklasse unsigned int ptri; /* Zeiger auf Information in Datei float xmin,ymin; /* orthogonale Extremwerte (min) float xmax,ymax; /* orthogonale Extremwerte (max) float xminl,yminl; /* orthogonale Extremwerte (min) lokal float xmaxl,ymaxl; /* orthogonale Extremwerte (max) lokal float x[4],y[4]; /* minimal umschreibendes Rechteck float xl[4],yl[4]; /* minimal umschreibendes Rechteck short del; /* Löschkennung short grpn; /* Gruppe, zu der Segment gehörte short layn; /* Layer, zu dem Segment gehört short attr; /* Vorrangattribut short lng; /* Länge der Information in Datei short npri; /* Anzahl der Primitiva im Segment short date[2]; /* Segmentdatum	*void own_mpur_sinf_(double elem[], int *nwas, int *iwas, int *gind, int *nr, int *typ);* *void own_mpur_sinfi(double elem[], int *nwas, int *iwas, int *iseg);* *void own_mpur_sind(double elem[], int *ifle, int *gind, int *nr, int *typ);* *void own_mpur_sbkz_(double elem[], int *ifile, int *iseg);*

Objekt	*Datenstruktur*	*Methode*
Primitivum	struct own_prim_hdr /* Headerinformation pro Primitivum {short typ; /* Typ short lng; /* Länge short attind; };/* Attributindex, zeigt auf gemeinsame Attr.	*char *own_mpur_prim_(double elem[], int *nwas, int *iwas, int *prim);* *char *own_mpur_prim_cache (double elem[], int *nwas, int *iwas, int *ipradr);*
Polylines Polymarker Polylines Fill area	struct own_prim_hdr o_h; /* Headerinformation short np; /* Anzahl Punkte O_KOO xy[NPOXY]; /* Koordinaten	*void own_mpur_02(double elem[], int *nwas, int *iwas, unsighned short int ipradr);*
allgemeiner Kommentar	struct own_prim_hdr o_h; short ni; float inf [NPINF];	*void own_mpur_06(double elem[], int *nwas, int *iwas, unsighned short int ipradr);*
Kreisstueck	struct own_prim_hdr o_h; short pad; short filken; /* Kennung, wie zu füllen ist O_KOO xy0; /* Mittelpunkt O_KOO xy1; /* Anfangspunkt O_KOO xy2; /* Endpunkt	*void own_mpur_07(double elem[], int *nwas, int *iwas, unsighned short int ipradr);*
Text	struct own_prim_hdr o_h; short nc; /* Anzahl der Zeichen im Text short bezug; /* Lage des Bezugspunktes short umrahm; /* Art der Umrahmung O_KOO xy; /* Position float angle; /* Schriftwinkel in Grad char txt [NPOXY]; /* Text selbst	*void own_mpur_04(double elem[], int *nwas, int *iwas, unsighned short int ipradr);*

Tabelle 5-2: Datenstruktur von Gruppe, Segment und Primitivum in bmf_-Datei

Die Identifizierung einer 2D-Grafikdatei erfolgt durch Auswahl aus der Dateiliste im Dialogfenster „Grafik platzieren“. Durch Doppelklick auf den Dateinamen oder Wählen von „Vorschau“ wird die ausgewählte Grafikdatei grafisch dargestellt (Bild 5-20).

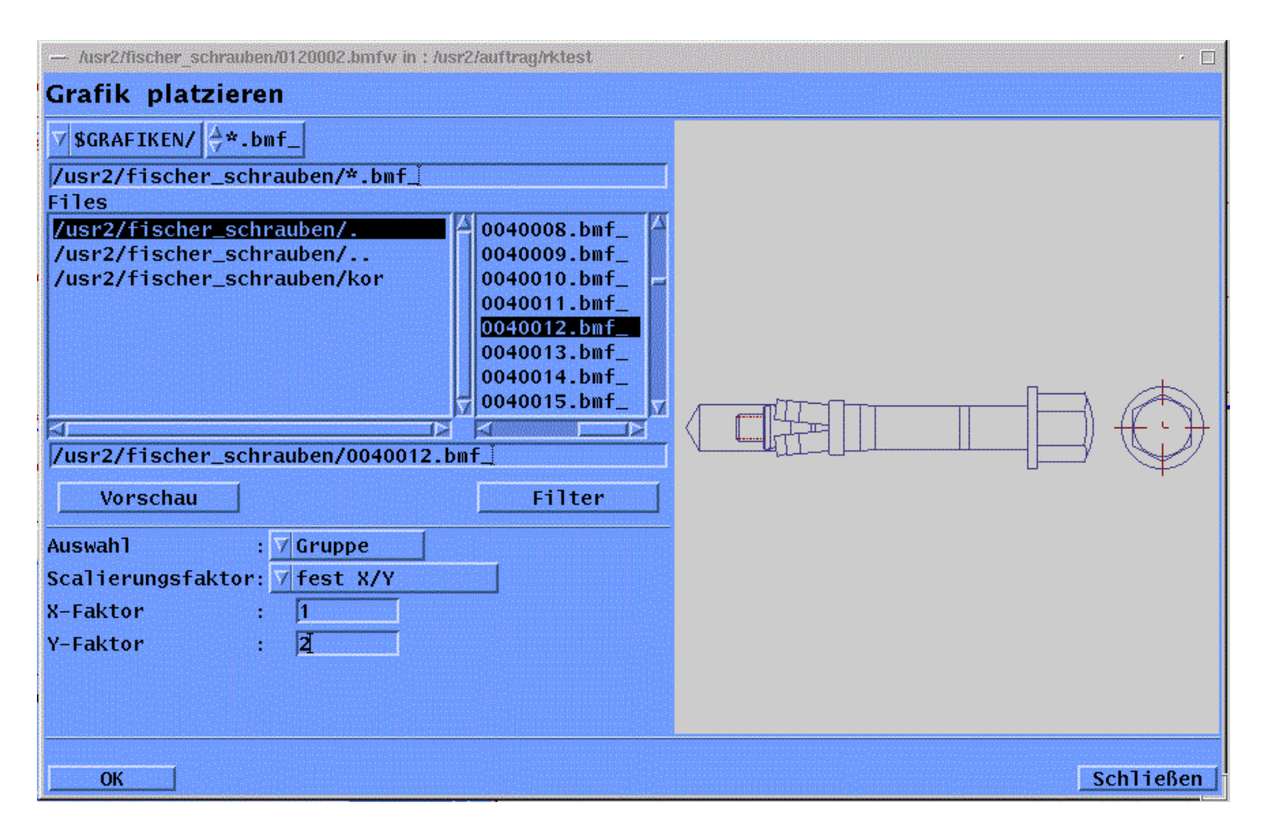


Bild 5-20: 2D-Grafik mit Vorschau zum Platzieren

Es wird in Bild 5-20 eine Auswahl an 2D-Dateien aus Kunden- oder BOCAD-Ordnern angeboten. Nur bmf_ Dateien werden hier gelistet und dann in der Vorschau dargestellt.

Vier Auswahlmöglichkeiten werden über Schaltfläche „Auswahl“ angeboten:

- Datei: Die ausgewählte Datei wird in die Zielzeichnung platziert.
- Gruppe: Die ausgewählten Gruppen werden in die Zielzeichnung platziert.
- Segment: Die ausgewählten Segmente werden in die Zielzeichnung platziert.
- Primitivum: Die ausgewählten Primitiva werden in die Zielzeichnung platziert.

Wird die Auswahlmöglichkeit auf **Datei** eingestellt, kann der Benutzer die Datei als eine Rootgruppe mit einem automatisch errechneten Skalierungsfaktor, der dem Original aus der Quelldatei, der Hauptgruppe oder der Zielgruppe entspricht, in die Zielgruppe platzieren. Das heißt, dass der Maßstab der Zielgruppe berücksichtigt

wird. Wenn zum Beispiel die Originaldatei einen Maßstab von 1:1 und die Zielgruppe einen Maßstab von 1:10 hat, wird die Grafik in der Größe um den Maßstab von 1:10 reduziert in die Zielgruppe platziert, falls der Benutzer **Maßstab: entsprechend Zielgruppe** für den Skalierungfaktor auswählt. Darüber hinaus kann der Benutzer einen Winkel eingeben, wenn die Grafik in der Zielgruppe gedreht werden soll. Nach dem Klicken von „OK“ wird der Benutzer aufgefordert, in der Vorschau einen Bezugspunkt zu picken, oder der Benutzer setzt bei der Wandlung DXF -> bmf_ einen Ankerpunkt voraus, der automatisch als Bezugspunkt für das Platzieren verwendet wird. Weiterhin erläutert die Vorschau Ausschnittvergrößerungen, wie in Abschnitt 4.2.6 beschrieben. Bei der Bestimmung der neuen Platzierung mit gedrückter linker Maustaste wird ein temporärer Rechteck-Rahmen dynamisch angezeigt. Dieser Rahmen ist genau so groß wie die später in der Zielgruppe erscheinende Grafik. Die 2D-Grafikdatei wird dann als Grafikelement am Platzierungspunkt in der Zeichnung platziert und kann als Gruppe, Segment oder Primitivum weiter kopiert, verschoben, skaliert, gespiegelt oder gedreht werden.

Wird die Auswahlmöglichkeit auf **Gruppe**, **Segment** oder **Primitivum** eingestellt, stehen folgende Skalierungsmöglichkeiten zur Verfügung:

- fest: Die ausgewählten Gruppen, Segmente oder Primitiva werden mit ei-nem festen Skalierungsfaktor (der Faktor ist in X- und Y-Richtung gleich) in die Zielzeichnung platziert. Der eingegebene Wert gilt für X- und Y-Skalierung.
- fest X/Y: Die ausgewählten Gruppen, Segmente oder Primitiva werden mit zwei festen Skalierungsfaktoren für jeweils X und Y in die Zielzeichnung platziert. Der eingegebene Wert im X-Feld gilt für X-Skalierung, im Y-Feld für Y-Skalierung.
- über Lasso: Die ausgewählten Gruppen, Segmente oder Primitiva werden mit zwei Skalierungsfaktoren für jeweils X und Y, die aus den Lasso-koordinaten errechnet werden, in die Zielzeichnung platziert. Das Lasso-Fenster definiert die Größe des Rahmens, mit der die Grafik in der Zielgruppe erscheinen soll.

- über (min) Lasso: Die ausgewählten Gruppen, Segmente oder Primitiva werden mit einem Skalierungsfaktor in die Zielzeichnung platziert. Ausgehend von den Lassokoordinaten wird nur der kleinere Skalierungsfaktor für die X/Y-Skalierung benutzt.
- über (max) Lasso: Die ausgewählte Gruppen, Segmente oder Primitiva werden mit einem Skalierungsfaktor in die Zielzeichnung platziert. Ausgehend von den Lassokoordinaten wird nur der größere Skalierungsfaktor für die X/Y-Skalierung benutzt.
- automatisch: Die ausgewählten Gruppen, Segmente oder Primitiva werden mit einem Skalierungsfaktor, der durch die Auswertung der Maßstäbe der Quellgruppe und der Zielgruppe ermittelt wird, in die Zielzeichnung platziert.

Die ausgewählten Gruppen, Segmente oder Primitiva können auch gedreht in die Zielzeichnung platziert werden. Die Rotation kann sich automatisch aus den gepickten Punkten ergeben. Das heißt, dass der Benutzer ein oder zwei Quellpunkte und ein oder zwei Zielpunkte picken kann, aus denen der Rotationswinkel ermittelt wird.

Für die Vorschau auf den zur Verfügung stehenden Vorrat an Ordnern mit 2D-Grafik wird die in Bild 5-21 dargestellte Lösung vorgeschlagen. Im Kopf der Vorschau gestattet ein Listenfeld, den Ordner des gewünschten Herstellers von Zulieferteilen auszuwählen. Es erscheinen dann wie eine Diaserie in der Reihenfolge der Bildnamen so viele 2D-Bilder, wie in den Fensterbereich passen. Die weiteren Bilder des Ordners erhält man über die vertikale Bildlaufleiste. Das gewünschte Bild wird mit der linken Maustaste in das Zielfenster des CAD-Systems gezogen.

Nach der Aufbereitung kann eine 2D-Grafik von jedem Anwender in wirklich einfacher Weise aus einer Vorschau mit den Artikelbezeichnungen des Herstellers ausgewählt und in die Zielzeichnung gezogen werden. Intelligente Leistungen bei dieser Aktion sind neben einer zweckmäßig organisierten Vorschau die automatische Skalierung der 2D-Grafik vom Ursprungsmaßstab und der Ursprungseinheit in den Zielmaßstab und die Zieleinheit sowie das Einrasten des Referenzpunktes auf den gewählten Zielpunkt.

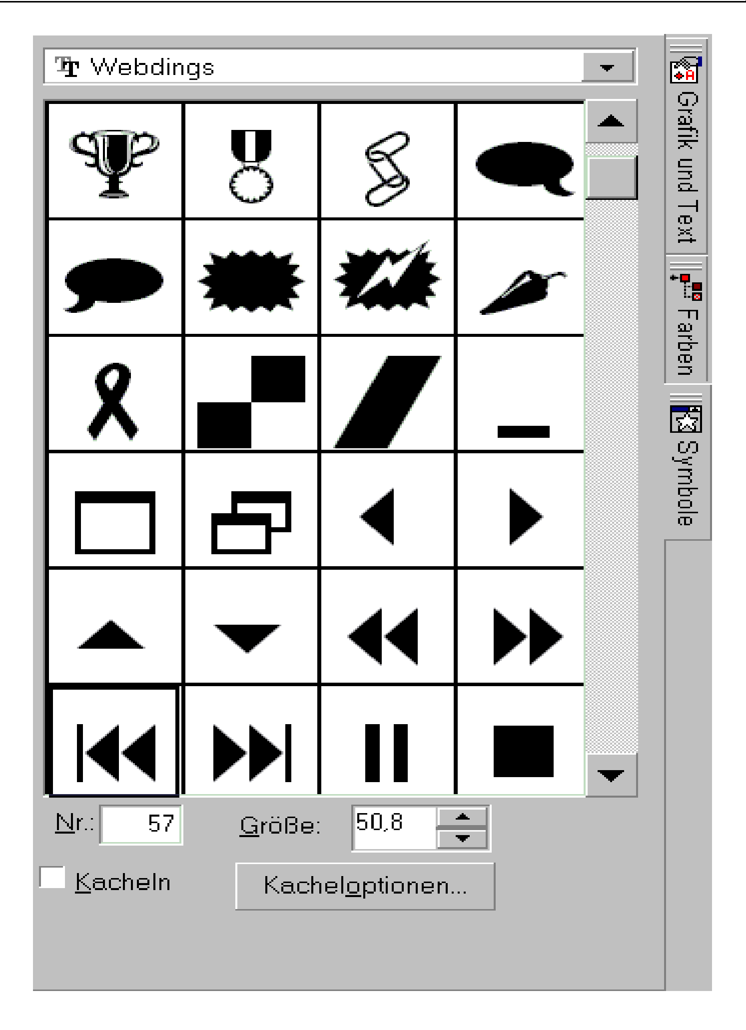


Bild 5-21: Vorschau eines Ordners mit 2D-Grafiken

Der Ordner der Vorschau soll wie in dem aus allen Microsoft Programmen bekannten Explorer grafisch wählbar sein. Nach Klick auf einen wählbaren Referenzpunkt im gewünschten Bild der Vorschau soll dieses Bild durch Klick auf einen Zielpunkt in der Zielzeichnung automatisch maßstabgerecht als ein Segment erscheinen. Der Arbeitspunkt soll auf dem Zielpunkt liegen. Das Bild ist am Drehgriff um den Referenzpunkt drehbar, da die Ursprungslage nicht immer ohne Drehung auch die notwendige Ziellage darstellt.

Als Direktaktionen mit Griffen an importierter 2D-Grafik sind nach den o. g. einheitlichen Methoden folgende Leistungen vorzusehen:

Drehen um den Referenzpunkt, Verschieben, Löschen, Kopieren und Spiegeln sowie Eigenschaften ändern.

Skalieren ist naturgemäß bei automatischer Maßstabumrechnung eine unerlaubte Aktion und darf daher nur nach Rückbestätigung der entsprechenden Warnung ausgeführt werden.

Führende Unternehmen im Bauwesen konstruieren selbst völlig individuelle Bauten im Detail systematisch, d. h. mit immer wiederkehrenden, gut durchdachten Lösungsdetails, die sich nur maßlich unterscheiden, nicht aber in der grundsätzlichen Bauart. Bei auftragsübergreifender Wiederholhäufigkeit lohnt es sich daher, 2D-Bilder von firmenspezifischen Detailbauarten in einer firmeneigenen 2D-Grafikbibliothek (Ordner) unter wählbarem Namen zu speichern. Für diese Aktion „2D-Grafik exportieren“ ist daher eine Nutzoberfläche und objektspezifische Intelligenz zu entwickeln, um z. B. bei laufender Konstruktion für Wiederverwendung eine geeignete Grafik in der firmeneigenen 2D-Grafikbibliothek zu speichern. Danach steht die eigene 2D-Grafik in exakt der gleichen Art wie die o. g. importierte Fremdgrafik für einfaches und effizientes Arbeiten allen Anwendern zur Verfügung.

5.2.7 Bauspezifische Symbole

Bauspezifische Symbole werden im Bauwesen als Hinweis auf Raster, Bauteile und Profile verwendet. Sie können sich auf ein bestimmtes Objekt oder mehrere bestimmte Objekte beziehen. Als Beispiel wird das folgende bauspezifische Symbol „Standardbezeichnung“ konzipiert und für das hier untersuchte CAD-System realisiert.

Bild 5-22 zeigt die allgemeine Form einer „Standardbezeichnung“, die am Ankerpunkt mit der Anrisslinie auf ein anderes Objekt zeigt. Falls sich das Symbol auf mehrere Objekte bezieht, zeigen entsprechend viele Anrisslinien dorthin (Bild 5-23), sofern die Objekte benachbart, d. h. in sinnvoller Nähe sind.

Bei der Realisierung des 2D-Grafiksymbols „Standardbezeichnung“ werden folgende Kriterien berücksichtigt:

- Das den Bezeichnungstext umrandende Symbol kann auf verschiedene Typen, beispielsweise Kreis, Rechteck oder Raute, eingestellt werden.
- Die Lage des Symbols kann bezüglich des Platzierungspunktes zu „oben“, „unten“,

„rechts“ oder „links“ durch Picken eines Seite-Punktes eingestellt werden.

- Zweizeiliger Text kann eingegeben und mit verschiedenen Attributen verbunden werden. Der Text kann sich dem Symbol „intelligent“ anpassen.
- Die Textattribute und der Textinhalt können editiert und durch anderen Text ersetzt werden.
- Linienattribute können durch ein Dialogfenster eingestellt werden.
- Das Symbol kann mit oder ohne Anriss erzeugt werden.
- Verschieben des Objekts „Standardbezeichnung“ mit Anriss sollte dadurch „intelligent“ durchgeführt werden, dass einerseits der Verschiebeanfasser den Knickpunkt der Anrisslinie bei festem Anker an einen neuen Ort bewegt, andererseits wird die Ankerlinie beim Ziehen des Ankeranfassers an einen neuen Zielort gummibandartig mitgezogen. Beim Verschieben des Objekts „Standardbezeichnung“ ohne Anriss wirkt diese Aktion wie ein normales Verschieben.

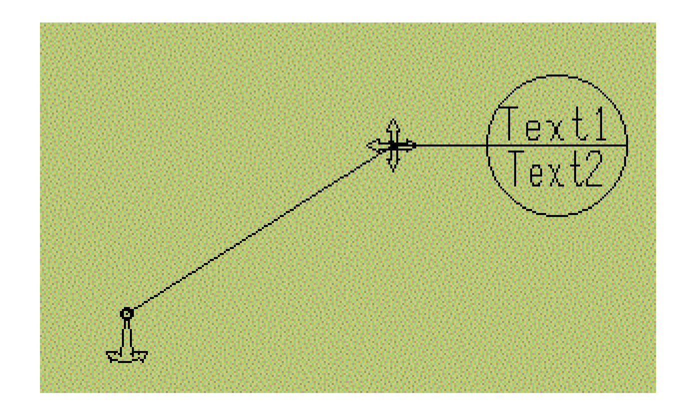


Bild 5-22: Anfasser für „Standardbezeichnung“ mit Anriss

Zu den Variablen einer „Standardbezeichnung“ mit Anriss zählen zwei Textattribute, Linienattribute, Anrisstyp und Platzierungspunkt. Tabelle 5-3 zeigt die Variablen und ihre Voreinstellung.

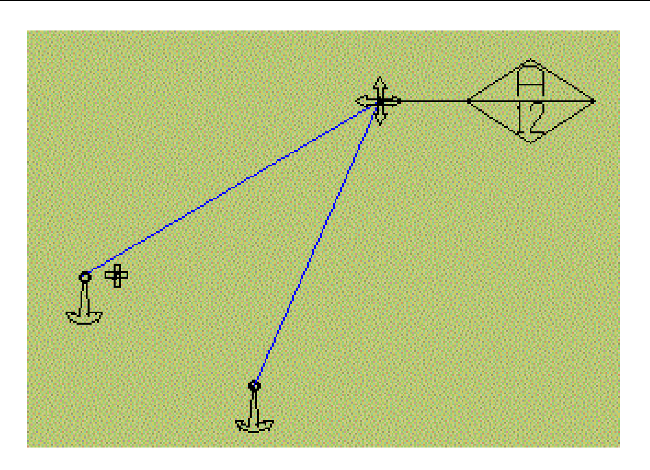


Bild 5-23: Mehrfachanriss durch <Strg>

Variable		***Mögliche Einstellung***	***Voreinstellung***
Textattribute /(2. Zeile)	Schrifthöhe	0.050, 0.075, 0.100, 0.125, ..., 200.000	0.350
	Strichstärke	Stift 1, Stift 2, Stift 3, ... ,Stift 9	Stift 1
	Breite/Höhe	0.10, 0.20, 0.30, ..., 10.00	1.20 /(0.60)
	Schriftneigung	90.0, 75.0, ISO	90.0
Textinhalt /(2. Zeile)		Text frei eingebbar	A /(12)
Ausrichtung		oben, unten, links, rechts	oben
Symbol-Rotationswinkel		0, 90	0
Linienattribut	Strichart	durchgezogen, gestrichelt, .., gepunktet	durchgezogen
	Rapport	0.875, 1.25, 1.75, 2.5, ... , 20.0	3.5
	Strichstärke	Stift 1, Stift 2, Stift 3, ... ,Stift 9	Stift 1
mit oder ohne Anriss		ohne Anriss, mit Anriss	mit Anriss
Symbolform		Kreis, Rechteck, Raute, Oval	Kreis
Breite des Symbolrahmens		Zahl frei eingeben	150
Höhe des Symbolrahmens		Zahl frei eingeben	100
Länge der Hilfslinie		Zahl frei eingeben	100
In Datenbank?		0/1	0
In Hauptgruppe?		0/1	0
Platzierungspunkt (Symbolort)		ermittelt durch Picken bei Generieren des Objekts	
Anrisspunkte		ermittelt durch Picken bei Generieren des Objekts	

Tabelle 5-3: Variablen und ihre Voreinstellungen des Objekts „Standardbezeichnung“

Die Identifizierung einer „Standardbezeichnung“ erfolgt durch Rechtsklick auf das Symbol. Zur Bestätigung erhält das Symbol in seinem Platzierungspunkt einen Verschiebeanfasser. Falls die „Standardbezeichnung“ mit Anriss erzeugt wurde, erhält die Anrisslinie in ihrem Knickpunkt einen Verschiebeanfasser. Wenn nicht die Anrisslinie, sondern der Text selbst geändert werden soll, ist der Text zu identifizieren, wie in Abschnitt 5.2.3 definiert.

Vier direkte Aktionen werden nach der Identifizierung möglich (neben dem Löschen durch <Entf>). Der Verschiebeanfasser bewegt den Knickpunkt der Anrisslinie bei festem Anker an einen neuen Ort. Das Symbol, der Text und die Hilfslinie bleiben dabei horizontal, die Linie zum Anker bewegt sich gummibandartig mit.

Die gleiche Aktion, jedoch gezogen bei gleichzeitig gedrückter Taste <Strg>, kopiert die „Standardbezeichnung“ einschließlich aller Eigenschaften in ihrer aktuellen Form an einen neuen Zielort.

Ziehen des Ankeranfassers an einen neuen Zielort zieht die Ankerlinie gummibandartig mit, alle übrigen Elemente bleiben unverändert.

Die gleiche Aktion, jedoch gezogen bei gleichzeitig gedrückter Taste <Strg>, erzeugt eine zusätzliche Ankerlinie mit Anker, so dass zwei Ankerlinien vom Knickpunkt ausgehen, Bild 5-23. Ebenso sind weitere Ankerlinien zum Symbol zu erzeugen.

5.2.8 Bemaßungen

Eine der wesentlichen Leistungen von CAD-Systemen der oberen Leistungsklasse ist die automatische Bemaßung von Objekten. Im untersuchten CAD-System wird deshalb z. B. die Bemaßung innerhalb von Werkstattzeichnungen ohne Zutun des Anwenders automatisch ausgeführt. Hierfür ließen sich weitgehend allgemeingültige und von der Praxis akzeptierte Bemaßungsregeln herleiten und programmtechnisch verwirklichen. Dieser Entwicklungs- und Forschungsprozess zu Bemaßungsregeln dauerte allerdings infolge der Komplexität der individuellen Stahl-, Glas-, Fassaden- und Holzbauobjekte mehr als zwei Jahrzehnte und ist auch heute noch nicht in allen praktisch auftretenden Sonderfällen vollständig abgeschlossen. Mit automatischen Werkstattzeichnungen und automatischer Rasterbemaßung erfolgt zwar die Hauptarbeit, etwa 80% des Bedarfsspektrums insgesamt, ohne Zutun des

Anwenders. In Übersichten und Klärungsblättern jedoch sind die Regeln und Gepflogenheiten derart unterschiedlich von Unternehmen zu Unternehmen, dass auf ergänzende interaktive Bemaßung für Kontrollmaße etc. nicht ganz verzichtet werden kann. Bemaßungen sind daher auch als intelligente Objekte so zu konzipieren, dass sie einfach und direkt erzeugt und ebenso modifiziert werden können.

Dazu wird im Folgenden das intelligente Bemaßungsobjekt entwickelt, Bild 5-24.

Aus den für alle Objekte spezifisch zu gestaltenden CAD-Methoden sind die in der folgenden Liste unterstrichenen Befehle für Bemaßungen relevant:

Erzeugen und Platzieren, Identifizieren, Informieren, Einrasten, Löschen, Eigenschaften (Attribute) Ändern, Verschieben, Drehen, Kopieren, Klonen, Spiegeln, Detaillieren, Editieren, Erweitern, Reduzieren, Verbinden (Gruppieren), Darstellen, Bezeichnen, Undo, Redo, Onlinehilfe.

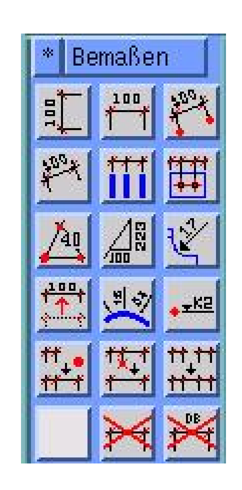


Bild 5-24: Werkzeuge „Bemaßen“, Werkbild BOCAD

Die Werkzeugleiste für das Erzeugen und Platzieren von Bemaßungen muss nach den Erfahrungen langjähriger CAD-Konstrukteure mindestens folgende Befehle als Symbole anbieten:

Bemaßung horizontale X-Richtung, vertikal Y-Richtung, 1 bis n Punkt bei freier Richtung, Teilegruppe, Schraubenbild, Winkel, Steigungsdreieck, Radius, Verschieben, Bogenbemaßung, Höhenkoten, Element hinzufügen, Element entfernen, Vereinigen.

Insgesamt sind, auch für seltenere Fälle, die in Bild 5-25 dargestellten Befehle für Bemaßungen erforderlich.

Aus den 2D-Zeichnungen oder 3D-Modellen der CAD-Systeme müssen Bemaßungen abgeleitet werden, um exakte Konstruktionsunterlagen für die

endgültige Fertigung, den Zusammenbau und die Montage zu erstellen. Alle CAD-Systeme bieten die Möglichkeit herkömmlicher 2D-Bemaßungsarten wie linearer Bemaßung (horizontale, vertikale, ausgerichtete und freie Bemaßung), Winkelbemaßung, Durchmesserbemaßung, Radienbemaßung etc.



Bild 5-25: Befehlsliste „Bemaßen“, Werkbild BOCAD

Das untersuchte CAD-System ist ein 3D-orientiertes Konstruktionssystem. Es besitzt Bemaßungsobjekte als sekundäre, darstellungsorientierte 2D-Funktionen. Obwohl alle Funktionen für herkömmliche 2D-Objekte grundsätzlich vorhanden sind, kann zusätzlich eine automatische Anpassung der Bemaßungsobjekte an das Höhenniveau im dreidimensionalen Raum eine schnellere und bequemere Arbeitsweise ermöglichen. Als Beispiel wird ein leistungsfähigeres Bemaßungsobjekt „Höhenkote“ konzipiert und realisiert, mit dem die interaktive Bemaßungsarbeit deutlich beschleunigt werden kann.

Die Höhenkote wird im Bauwesen für die Bemaßung der Höhe verwendet. Folgende Anforderungen werden an das Objekt bei der Entwicklung gestellt:

- Alle Parameter werden gemäß DIN 1356 und praktischer Anwendung als Voreinstellung (engl. default setting) eingestellt.

- Die Maßzahl wird auf das NULL-Niveau (Spezifikation: „globales NULL-Niveau“) bezogen und automatisch bei der Bemaßungsgenerierung ermittelt.
- Symboltyp und die Symbolgröße des Kote-Symbols können wahlweise „offen“ oder „gefüllt“ eingestellt werden.
- Zusätzlicher Text kann eingegeben und mit der Maßzahl verbunden sowie nachträglich editiert werden.
- Textattribute und die Position bezüglich des Symbols können gesteuert werden.
- Linienattribute bzw. der Abstand zwischen Maßhilfslinie und Maßpunkt können eingestellt werden.
- Verschiebungen der Höhenkote beziehen sich nur auf Bemaßungspunkte und können nur horizontal durchgeführt werden.

Bild 5-26 zeigt die allgemeine Form des Objekts „Höhenkote“.

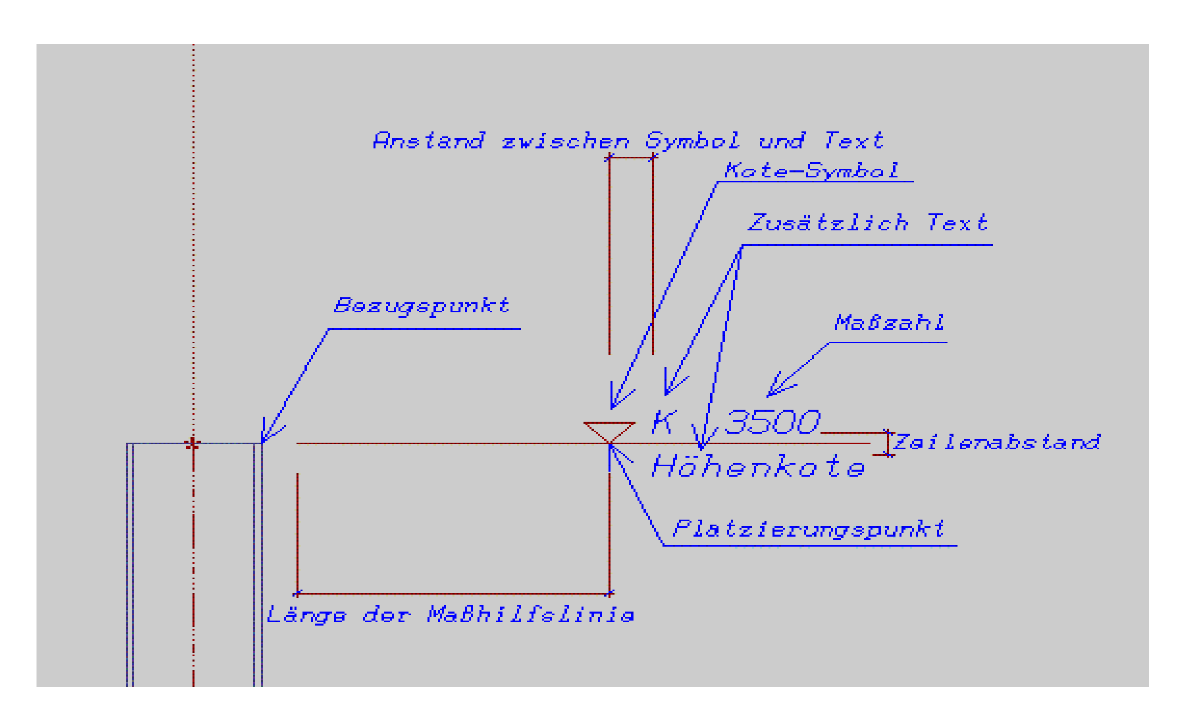


Bild 5-26: Bemaßungsobjekt - Höhenkote

Die dazu gehörenden Variablen und ihre Voreinstellungen stellt Tabelle 5-4 dar.

Variable		***Mögliche Einstellung***	***Voreinstellung***
Linienattribute	Strichstärke	Stift 1, Stift 2, Stift 3, ... , Stift 9	Stift 1
	Strichart	durchgezogen, gestrichelt, ..., gepunktet	durchgezogen
	Rapport	0.875, 1.25, 1.75, 2.5, ... , 20.0	3.5
Textattribute	Schrifthöhe	0.050, 0.075, 0.100, 0.125, ..., 200.000	0.350
	Umrandung	keine, unterstrichen, ..., opaque	opaque
	Strichstärke	Stift 1, Stift 2, Stift 3, ... , Stift 9	Stift 2
	Breite/Höhe	0.10, 0.20, 0.30, ..., 10.00	0.90
	Schriftneigung	90.0, 75.0, ISO	75.0
Symbol-attribute	Höhe	Zahl frei eingeben	2.5
	Breite/Höhe	Zahl frei eingeben	2
	Richtung	Höhensymbol nach oben oder unten	nach oben
	Symboltyp	offen, voll, links oder rechts gefüllt	offen
Textlage	Abstand zu Sym.	Zahl frei eingeben	2.5
	Zeilenabstand	Zahl frei eingeben	1.5-fach
Länge der Maßhilfslinie		punktbezogen, minimale, feste Länge	punktbezogen
Maßzahl		automatisch ermittelt bei Generieren des Objekts	
1. zus. Text	nötig?	0/1	0
	Inhalt	Text frei eingeben	k
2. zus. Text	nötig?	0/1	0
	Inhalt	Text frei eingeben	
Maßeinheit		mm, cm, feet/inch, m	mm
Genauigkeit der Maßzahl		Zahl frei eingeben	1
NULL-Niveau		global oder lokal	1 (global)
In Hauptgruppe?		0/1	0
In Datenbank?		0/1	0
Bezugspunkte		ermittelt durch Picken bei Generieren des Objekts	
Platzierungspunkt		ermittelt durch Picken bei Generieren des Objekts	

Tabelle 5-4: Variablen und ihre Voreinstellungen der Höhenkote

Die Identifizierung einer Höhenkote erfolgt durch Rechtsklick auf das Kote-Symbol. Die Höhenkote bietet in ihrem Platzierungspunkt einen horizontalen Verschiebeanfasser an (siehe Bild 5-27).

Folgende direkte Aktionen am Objekt „Höhenkote“ werden dadurch möglich: Löschen durch <Entf>, Verschieben horizontal, Editieren des Textes sowie der Eigenschaften der Linie, des Textes und des Symbols.

Der Verschiebeanfasser bewegt Kote-Symbol, Text und Maßzahl an einen neuen horizontalen Ort, aber der Bezugspunkt bzw. der Anfangspunkt der Linie ändert sich nicht, da die Höhenkote sich sowohl beim Generieren als auch beim Verschieben „intelligent“ auf den Bezugspunkt bezieht.

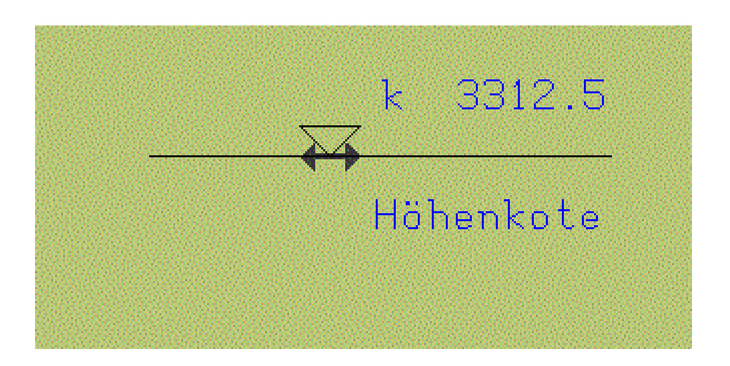


Bild 5-27: Anfasser für Bemaßungsobjekt - Höhenkote

5.3 Dreidimensionale Objekte

5.3.1 Raster-Objekte mit Achsen, Reihen und Koten

Bild 5-28 zeigt zur Klärung der Begriffe die einfachste Form eines quaderförmigen Rasters mit Achsen, Reihen und Höhenkoten.

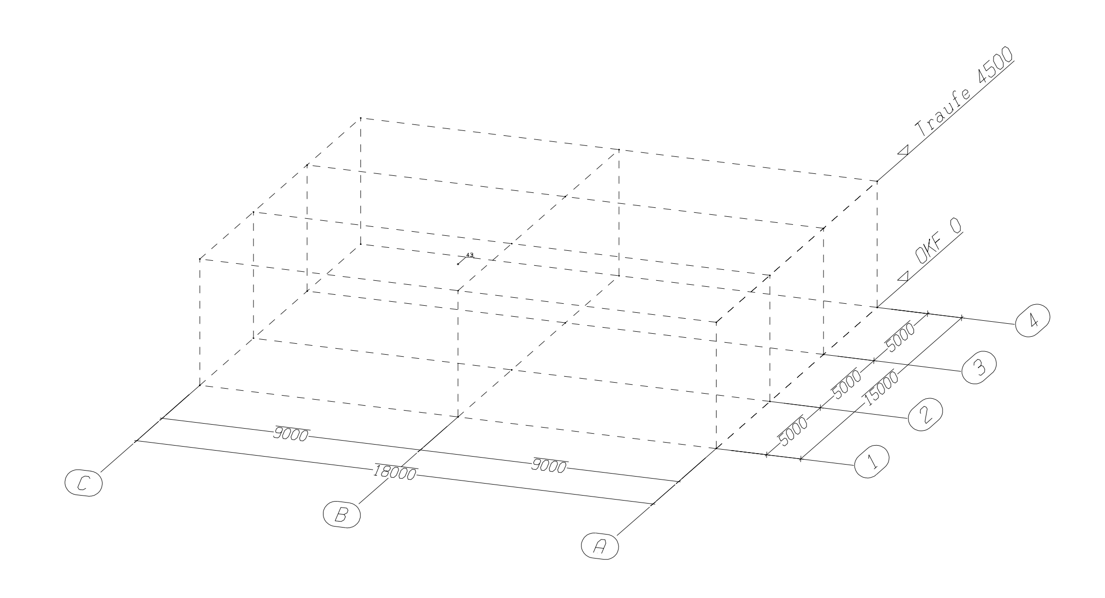


Bild 5-28: Quader als einfachste Form eines Rasters

Betrachtet man Raster im Prinzip als nichts Anderes als räumliche Stabwerke mit gleichartiger Bedeutung, halbiert sich für den Anwender der Lernaufwand für die Bedienung. Alle von Stäben ohnehin bekannten Operationen sind dann ebenso auf Elemente des Rasters anzuwenden. Die unterschiedliche Darstellung von Stäben und Rasterlinien einschl. Rasterbezeichnungen kann durch entsprechende Intelligenz des CAD-Systems übernommen werden, so dass der Anwender davon unberührt bleibt. Selbst parametrisierte Makros für häufiger genutzte Rasterformen, z. B. Hallen mit Giebeldach oder Sheddach, Treppentürme, sternförmige Raster, Kombination mehrerer Rastertypen etc., sind bei diesem objektorientierten Ansatz ebenso zu behandeln wie Makros für Stabwerke. Zumindest für Raster-Makros wurde in einem der AutoCAD basierten CAD-Systeme für Stahlbau, ProStahl 3D ein

ähnlicher, objektorientierter Ansatz verwirklicht, Bild 5-29.

Durch die Analogie zu Stäben haben Rasterebenen bzw. die darin enthaltenen Rasterlinien die gleichen Eigenschaften und Methoden wie Stäbe mit einigen Vereinfachungen. Rasterlinien bleiben stets Linien, werden also nicht durch ein Profil verkörperlicht, so dass die entsprechenden Körperaktionen entfallen, z. B. das Ändern des Profils oder volumenverändernde Detaillierungsmethoden. Damit bleiben für Rasterlinien die in Bild 5-29 dargestellten, von Stäben bekannten Funktionsgriffe sinnvoll.

ProStahl 3D bietet die Möglichkeit, beliebige Arbeitsbereiche zu definieren. Es stehen Ihnen rechteckige, zylindrische, keil- und pyramidenförmige Arbeitsbereiche zur Verfügung.
Diese erleichtern die Orientierung im 3D-Raum erheblich, da sie zum einen die grundsätzlichen Systemmaße (z.B. Achsmaße) als Konstruktionshilfsobjekte darstellen können, zum anderen über die Definition des Arbeitsbereiches automatisch die dazugehörigen Ansichten (Vorderansicht, Seitenansicht, Draufsicht, etc.) erzeugt werden.
Für Sie als Anwender reduziert sich der Ansichtenwechsel somit auf einen einfachen Mausklick.
Der Arbeitsbereich stellt ein eigenes Objekt dar, dessen Parameter (z.B. Dachneigung, Achsabstand, Aufteilung, etc.) nachträglich geändert werden können.

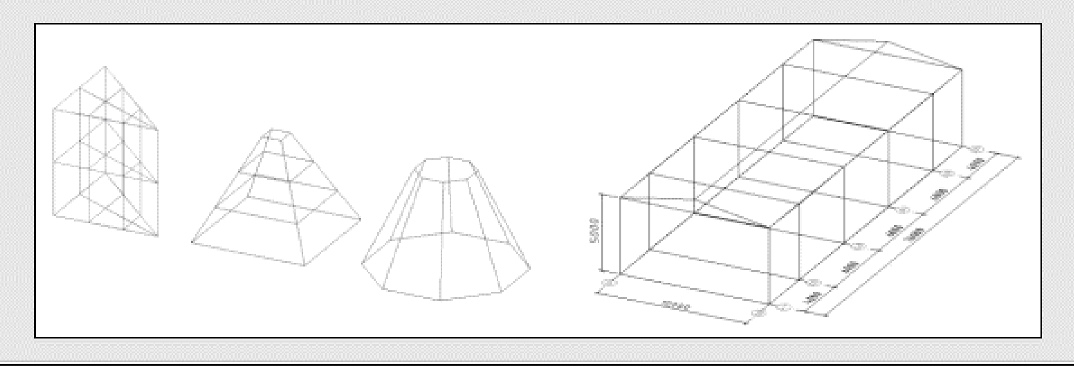


Bild 5-29: Objektorientierter Ansatz für Raster (Quelle ProStahl 3D)

5.3.2 3D-Körper Profilstäbe, Bleche, Kantteile, Baugruppen

Das Erzeugen und Platzieren von Teilen, d. h. einzelnen Profilstäben, Blechen und Kantteilen, ist nach der Rasterfestlegung zu Beginn der Konstruktionsarbeit die häufigste Aktion. Der entscheidende „erste Eindruck“, ob ein CAD-System einfach und effizient zu bedienen ist, wird gerade durch diese Aktionen bestimmt. Hier muss also in besonderem Maße ingenieurwissenschaftliche Forschung ansetzen, um positive Wirkung auf die Baupraxis zu erzielen.

Das Erzeugen von üblichen Teilen ist durch genormte Profilnamen bzw. selbsterklärende Namen, z. B. BL10*1500, recht einfach. Genormte oder lieferanten-spezifisch standardisierte Profile sind durch eine alphabetisch sortierte Auswahlliste wählbar. Selbst bei einigen tausend Profilen der Auswahlliste bleibt der Auswahlaufwand gering, da über die Anfangsbuchstaben der allgemein bekannten Profilnamen eine gezielt reduzierte Vorauswahl angeboten wird. Berufserfahrene CAD-Konstrukteure werden dennoch mit einer direkten Eingabe des Profilnamens in ein Eingabefeld schneller sein.

Für die Erzeugung von neuen Sonderprofilen, die im CAD-System noch nicht enthalten sind, wird eine integrierte Erfassungsmethode für die Querschnittsgeometrie und die Attribute angeboten. Nach der Erfassung stehen derartige Sonderprofile sowohl hinsichtlich der Nutzoberfläche als auch der auf diese Profile anwendbaren Methoden wie Normprofile zur Verfügung. Um zunächst das Verfahren für im System enthaltene Normprofile durchgängig darstellen zu können, wird die Erfassung von Sonderprofilen erst am Ende dieses Abschnitts erläutert.

Teile sind neben der Querschnittgeometrie auch durch Attribute gekennzeichnet, z. B. Benennung, Werkstoff, Oberflächenbeschichtung (getrennt nach Vorder- und Rückseite), klassifizierendem Positionsnummernbereich und Lageattribute wie Querlage des Profilstabes rechts, mittig oder links zu einer Funktionskante sowie seine Tiefenlage vor, mittig oder hinter der Arbeitsebene, jeweils mit möglichen Zusatzverschiebungen. Die Attribute, insbesondere die Lageattribute, sind ein wesentliches Mittel zur Vereinfachung der bauspezifischen Platzierung von Teilen und Beibehaltung der funktionsgemäßen Lage auch bei späterem Auswechseln gegen größere oder kleinere Profile. Das Problem, drei Freiheitsgrade der Rotation und drei Freiheitsgrade der Translation im Raum festzulegen, ist nämlich an und für sich auch bei gutem räumlichen Vorstellungsvermögen von CAD-Konstrukteuren prinzipiell nicht einfach. Erst konsequente programmtechnische Nutzung der Besonderheiten der funktionellen Platzierung von Teilen im Stahl-, Metall- und Holzbau erlaubt entscheidende Vereinfachungen der Nutzoberfläche. Nach Auswertung des Systemvergleichs verschiedener CAD-Systeme des Bauwesens und gängiger Standardsoftware wird daher im Folgenden eine einheitliche, effiziente und anschauliche Nutzoberfläche und Methode für das Erzeugen und Platzieren von

Teilen vorgeschlagen.

Um ein Teil zu erzeugen und zu platzieren, wird das Teile-Symbol aus der Symbolleiste auf den Punkt der Arbeitsebene des gewählten Bildschirmfensters gezogen, der Anfangspunkt des Profilstabes werden soll. Bestätigt man jetzt bereits durch Doppelklick oder die Eingabetaste, erscheint ein Dialogfeld im Modus „Stütze senkrecht zur Arbeitsebene“. Wählt man hingegen einen zweiten, weiteren Punkt als Endpunkt des Profilstabes, erscheint das Dialogfeld im Modus „Träger parallel zur Arbeitsebene“. (Wählt man noch einen dritten, nicht auf einer Geraden zu den beiden vorigen Punkten liegenden Punkt, so wird der Träger parallel zu dieser durch drei Punkte definierten Ebene im Raum platziert.)

Das einheitliche, modusgesteuerte Dialogfeld mit aktiver, dynamischer Grafik ist entsprechend Bild 5-30 gestaltet.

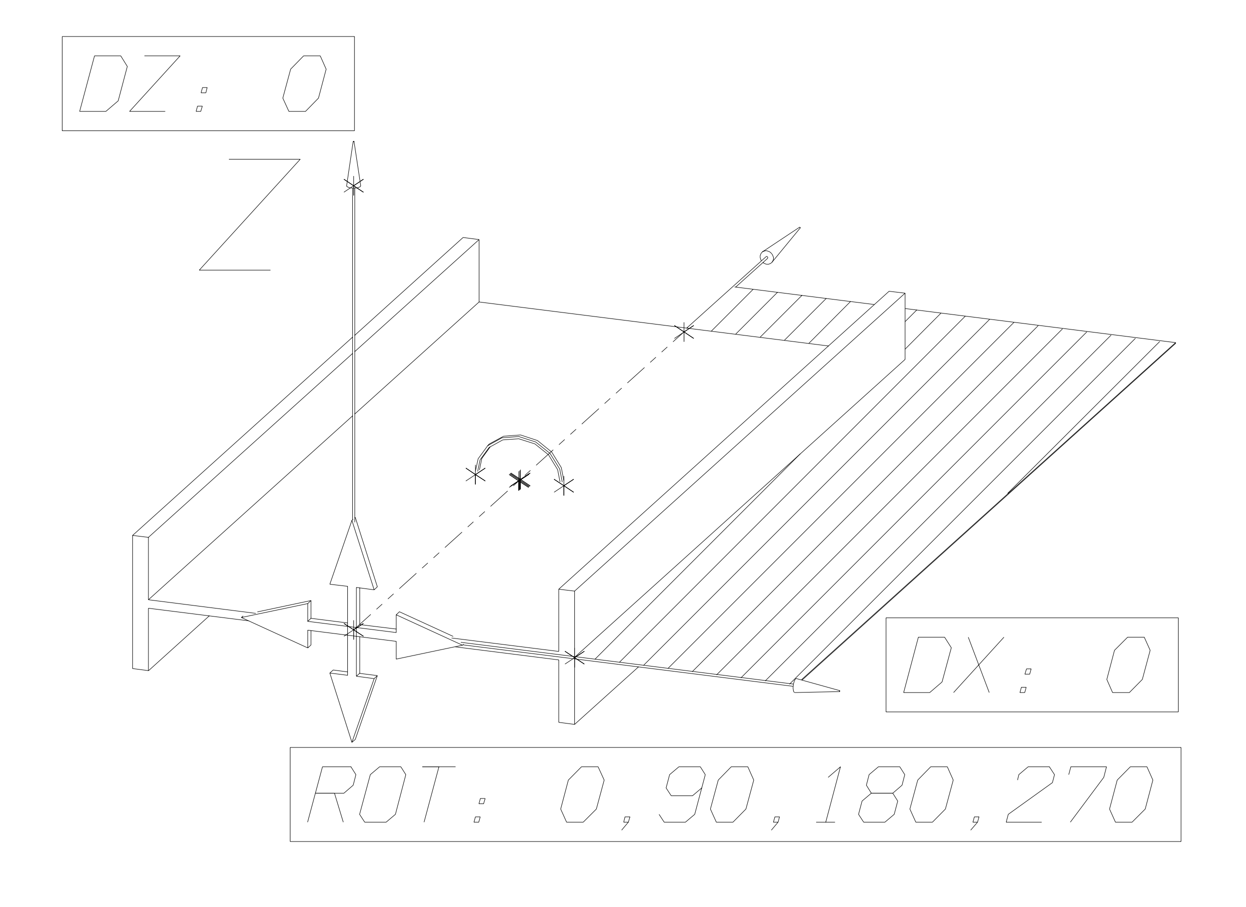


Bild 5-30: Dialogfeld „Teil erzeugen und platzieren“

Das letztgenutzte bzw. neu gewählte Profil wird sofort nach Eingabe des Profil-

namens mit den letztgenutzten Attributwerten und Lageparametern in Bezug zur Arbeitsebene im Dialogfeld und in allen offenen Fenstern anschaulich dargestellt (sofern es dort innerhalb der Sichttiefe liegt und die Darstellungsregeln entsprechen). Die Darstellung im Dialogfeld enthält mehrere Griffe. Mit Klick auf den Rotationsgriff wird das Teil um seine Längsachse gedreht und zwar so, dass die letztgenutzte funktionale Quer- und Tiefenlage dabei beibehalten wird. Der Verschiebegriff gestattet mit einer einzigen Ziehbewegung sowohl die Querlage als auch die Tiefenlage einzustellen, wobei das Teil automatisch an den Funktionskanten unter Berücksichtigung gewählter Zusatzverschiebungen einrastet. Im Stützenmodus wird zusätzlich das numerische Eingabefeld „Länge“ aktiv und ausgewertet. Die grafische Darstellung erfolgt im Stützenmodus als Stütze senkrecht zur Arbeitsebene, im Trägermodus als Träger parallel zur Arbeitsebene vom erstgepickten Startpunkt „1“ zum zweitgepickten Zielpunkt „2“.

5.3.3 Schraubverbindungen, Befestigungsteile, Dübel

Die Gesamtzahl von Schraubverbindungen, Befestigungsteilen und Dübeln eines Stahltragwerks übertrifft die Anzahl an Profilstäben und Blechen meist erheblich. Das Konstruieren dieser Verbindungselemente ist daher ein wesentlicher Faktor für die Gesamteffizienz eines CAD-Systems. Intelligente CAD-Methoden für die im Stahlbau gebräuchlichen Anschlussarten, z. B. Doppelwinkel, Kopfplatten, Rahmenecken oder Laschenstöße, konstruieren deshalb nicht nur alle Anbauteile des Anschlusses automatisch, sondern auch alle zugehörigen Schraub- und Schweißverbindungen. Gerade architektonisch anspruchsvolle Entwürfe erfordern aber häufig Anschlüsse individuell zu gestalten, d. h. auf übliche CAD-Methoden ist nicht zurückzugreifen. Für individuelle Gestaltung sind also Hilfsmittel zu konzipieren, die das Konstruieren von Schraubverbindungen ohne Lernaufwand einfach und anschaulich gestalten.

Bei Schweißverbindungen, Abschnitt 5.3.4, genügt es, die Partnerteile der Schweißverbindung zu wählen. Alles Weitere, also lagerichtige Anordnung einer Kehlnaht rings um die Umrandung der Kontaktfläche, ggf. unter Berücksichtigung unterschiedlicher Nahtdicken an Steg und Flansch, erledigt die CAD-Methode „Automatisch Schweißen“. Entsprechende Vereinfachungen übertragend wird die Nutzoberfläche für Schraubverbindungen gestaltet.

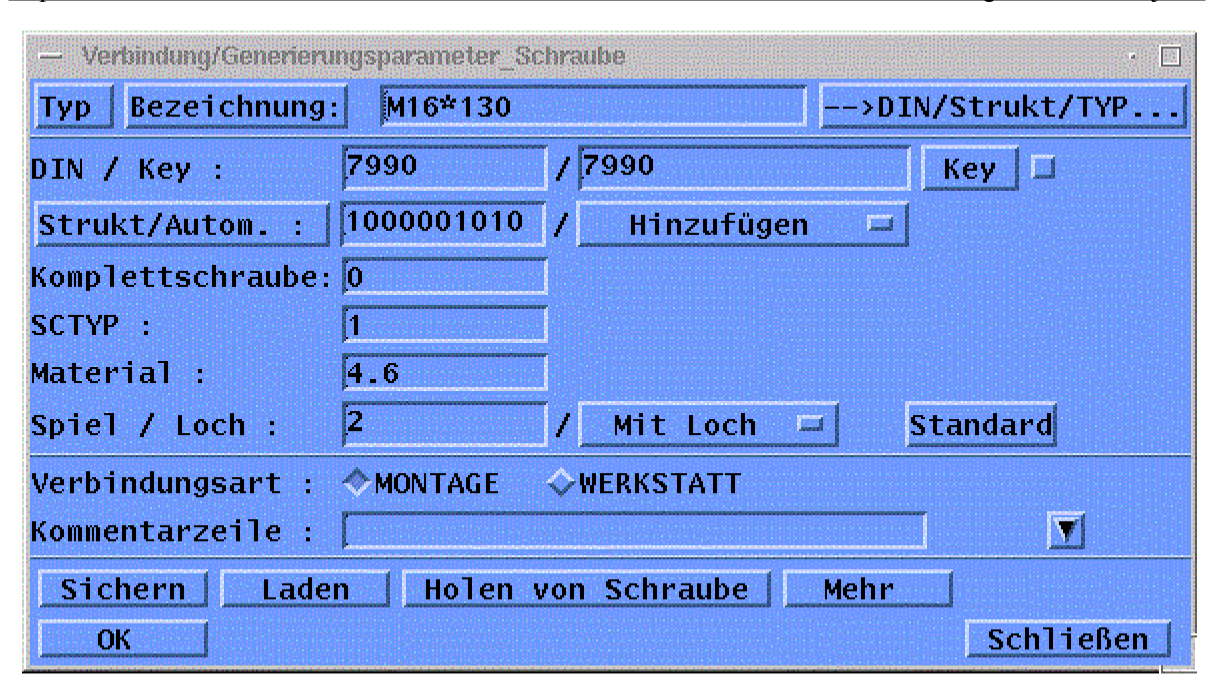


Bild 5-31: Schrauben-Eigenschaftsfenster

Zum **Erzeugen einer Schraubverbindung** sind zunächst die Verbindungspartner zu wählen, die senkrecht zu ihrer Kontaktfläche durch Schrauben verbunden werden sollen. Im Schrauben-Eigenschaftsfenster (Bild 5-31) wird dabei eine Liste der zuletzt verwendeten Schraubenarten zur Wahl angeboten, um den CAD-Konstrukteur zu erinnern, sich auf möglichst wenige Arten zu beschränken. Ist dennoch eine neue Art notwendig, bietet das Eigenschaftsfenster ähnlich wie bei Profilen über die Anfangszeichen des Schraubennamens eine gezielte Auswahl an. Bei Schraubenkatalogen mit mehreren tausend Arten bleibt die Auswahl somit effizient. Die gewählte Art wird mit dynamischer Grafik im Eigenschaftsfenster als Garnitur dargestellt, d. h. mit Unterlegscheibe(n), Keilscheibe, Mutter, Kontermutter und Sicherung, Bild 5-32. Über Kontrollfelder kann die artabhängige übliche Garnitur für Sonderfälle modifiziert werden. Die gewählte Garnitur wird als anschauliche Grafik quittiert, wobei die gewünschte Steckrichtung dargestellt wird, d. h. Schraubenkopf oben bzw. unten. Zur Information erscheint der nach Norm erforderliche Mindest-Randabstand der Schraube und der Mindestabstand zwischen zwei Schrauben.

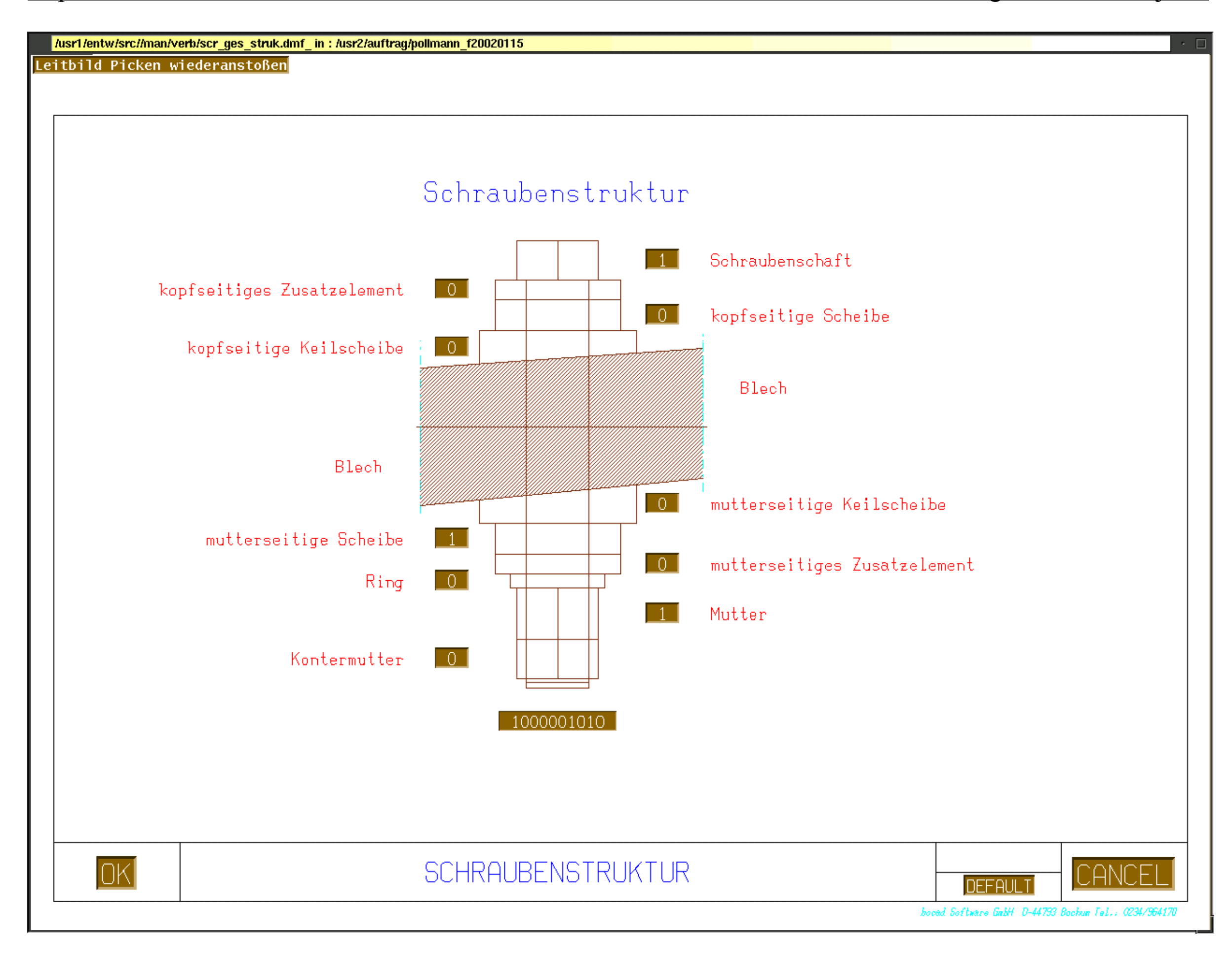


Bild 5-32: Leitbild „Schraubenstruktur", Werkbild BOCAD

Der technisch sinnvollen Platzierung von Schrauben liegt eine allgemein gültige Logik zugrunde, die im Folgenden zur Automatisierung genutzt wird. Die Verbindungspartner bieten im Bereich ihrer gemeinsamen Kontaktfläche dazu Hilfslinien an. Profilstäbe erzeugen auf ihren Reißmaßen (w-Maße) entsprechend den Werten ihrer Profiltabelle Hilfslinien, Blechkonturen erzeugen eine um den erforderlichen Randabstand der gewählten Schraube nach innen versetzte Hilfskontur. An den Schnittpunkten der Hilfslinien platzieren sich - sofern der Zwischenabstand eingehalten ist - Schrauben, die sich ihre Schraubenlänge selbst ermitteln. Die Distanz vom ersten zum letzten Durchstoßpunkt der Schraubenachse durch die Verbindungspartner ergibt die Klemmlänge. Zur Klemmlänge werden der Überstand und entsprechend der gewünschten Garnitur die Dicke der Unterlegscheibe(n), der Mutter usw. addiert. Diese theoretische Länge wird auf die nächstverfügbare Schraubenlänge des Kaufkatalogs aufgerundet. Bei Hohlprofilen, einem Sonderfall, kann der CAD-Konstrukteur das Prinzip der Hohlprofil-

Klemmlänge über ein Kontrollfelder festlegen, nämlich ob die Schraube innerhalb des Hohlprofils in unmittelbarer Nachbarschaft der Kontaktfläche enden oder vollständig durch das Hohlprofil bis auf die gegenüberliegende Außenseite durchgehen soll.

Bei sinnvoller Wahl des Schraubendurchmessers werden durch die geschilderte automatische Vorgehensweise nur selten überflüssige Schrauben erzeugt, die sich ggf. leicht mit Taste <Entf> löschen lassen.

Zur Modifikation vorhandener Schraubenanordnungen, Schraubenbilder genannt, bietet jede Schraube den Verschiebe- und Kopiergriff an, wie von Punktobjekten bekannt. Verschieben, Kopieren, Spiegeln, Löschen, Eigenschaften ändern (Schraubenart, Werkstoff, Garnitur, Steckrichtung, Kommentar, Hohlprofil-Klemmlänge) ist für Schraubenobjekte somit direkt, also durch bekannte Griffe am Objekt, durch die Taste <Entf> und das objektspezifische Eigenschaftsfenster ermöglicht.

5.3.4 Schweißverbindungen

Die Größenordnung der Anzahl von Schweißverbindungen üblicher Stahltragwerke liegt in grober Näherung immerhin bei der Hälfte der Teilezahl von Profilstäben und Blechen. Zudem wäre die Gestalt des oft räumlichen Polygonzugs einer Schweißnahtfolge und die Anordnung der Nahtflanken im Raum potentiell eine außerordentlich arbeitsaufwendige und fehleranfällige Aktion, wenn sie der CAD-Konstrukteur handwerklich interaktiv vornehmen müsste.

Für das **Erzeugen von Schweißverbindungen** ist daher automatisches Schweißen verwirklicht. Es genügt, wenn der CAD-Konstrukteur die Partnerteile der Schweißverbindung wählt und ggf. im Naht-Eigenschaftsfenster (Bild 5-33) für stegseitige und flanschseitige Nähte die gewünschte Nahtdicke festlegt. Alles Weitere, also lagerichtige Anordnung einer Kehlnaht rings um die Umrandung der Kontaktfläche, ggf. unter Berücksichtigung unterschiedlicher Nahtdicken an Steg und Flansch, erledigt die CAD-Methode „Automatisch Schweißen“.



Bild 5-33: Naht-Eigenschaftsfenster

Verschieben, Kopieren, Spiegeln, Löschen, Eigenschaften Ändern (Nahtart, Werkstoff, Stegnahtdicke, Flanschnahtdicke, unterbrochene Naht, Kommentar) ist für Schweißnahtobjekte somit direkt, also durch bekannte Griffe am Objekt, durch die Taste <Entf> und das objektspezifische Eigenschaftsfenster ermöglicht.

5.3.5 CAD-Methoden mit dynamischen, aktiven Leitbildern für Baugruppen und Anschlüsse

CAD-Methoden, die gestellte Konstruktionsaufgaben dank ihrer bauspezifischen Intelligenz selbständig und normgerecht lösen, sind ein entscheidendes Leistungsmerkmal von CAD-Systemen der obersten Leistungsklasse. Dazu stellen CAD-Systeme einen Vorrat an Konstruktionsmethoden oder Makros bereit, mit dem sich wiederkehrende Details automatisch konstruieren lassen. Der CAD-Konstrukteur muss am Bildschirm bei Einsatz einer CAD-Methode lediglich die Aufgabenstellung definieren und, nachdem die Aufgabe durch die entsprechende CAD-Methode praktisch verzögerungsfrei gelöst wurde, das dargestellte Ergebnis

beurteilen. Das Ergebnis kann dann bei Bedarf noch interaktiv modifiziert werden mit allen bisher vorgestellten Möglichkeiten der durch die Methode erzeugten Objekte.

CAD-Methoden werden aber nur dann von CAD-Ingenieuren genutzt, wenn die Definition der Konstruktionsaufgabe einfach und selbsterklärend ist, also nicht den Vorteil der automatischen Lösung durch den vorab erforderlichen Eingabe- und Lernaufwand mehr als aufzehrt. Die CAD-Methoden müssen deshalb so effizient und selbsterklärend sein, dass man ohne Schulung sofort damit arbeiten kann. Bei CAD - Konstruktionsmethoden mit sehr großem Variationsspektrum kann allerdings die Anzahl der Parameter so unüberschaubar groß werden, dass sie wegen zu langer Einarbeitungsdauer in der Praxis nicht genutzt werden.

Die führenden CAD-Systeme versuchen auf unterschiedliche Weise diese Problematik zu lösen. In einer diese Arbeit unterstützenden Diplomarbeit [12] wurde daher untersucht, welche der Gestaltungsweisen und Gestaltungselemente hier zukunftweisend und vorteilhaft sind. Mit diesen Grundlagen wurde dann ein optimal erscheinender Vorschlag einer direkten, textfreien CAD-Nutzoberfläche an einem konkreten Beispiel erarbeitet. In der Aufgabenstellung dieser Arbeit war folgendes Ziel formuliert:

„Es ist speziell zu untersuchen, mit Leitbildern gleichzeitig Online-Dokumentation und Eingabemedium zu vereinen. Leitbilder gestatten, direkt in die Bemaßung der dargestellten Anschlussbauart die gewünschten Maße einzugeben sowie Profile, Optionen und Werkstoffe zu wählen. Die bildliche Darstellung einer Bauart im Leitbild soll so anschaulich konzipiert werden, dass auf ergänzende textförmige Erläuterungen völlig verzichtet werden kann. Die Leitbilder sind dann ohne Übersetzung in allen fremdsprachigen, globalen Märkten einsatzfähig. Angesichts des sehr raschen Fortschritts von CAD-Systemen mit entsprechend häufigem Versionswechsel ist im globalen Wettbewerb die Vermeidung des ständigen Übersetzungsaufwands in alle Sprachen wichtig.“

Die Leistungsfähigkeit von Leitbildern als Online-Dokumentation und gleichzeitig Eingabemedium auch für weltweiten Einsatz wurde in [12] grundsätzlich belegt mit dem Hinweis, dass die Beschränkung auf aktive, aber nicht dynamische Grafik zwar hinreichend ist, die rasche Verständlichkeit aber durch zusätzliche dynamische

Darstellung der realen Situation nochmals verbessert werden kann. Die statische Grafik hat den erhaltenswerten Vorteil, dass bei der Gestaltung und den Größenverhältnissen darauf geachtet werden kann, dass die Lesbarkeit und Übersichtlichkeit optimal ist, gemessen an der aufgabenbedingten Fülle der Informationen. Ein typisches Beispiel des aktiven, statischen Leitbilds für die automatische Konstruktion einer kompletten Rahmenecke zeigt Bild 5-34.

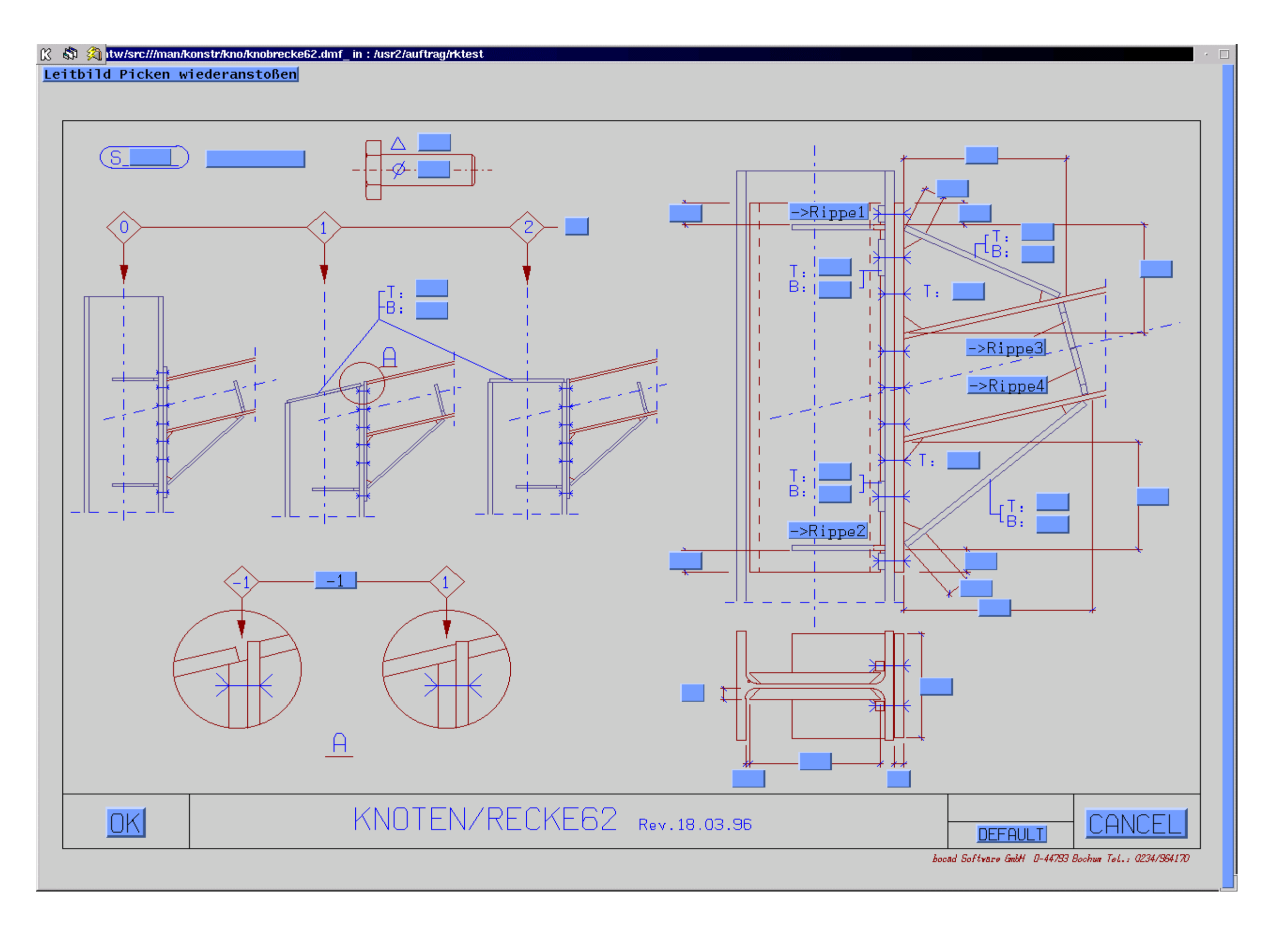


Bild 5-34: Leitbild „Rahmenecke-Eigenschaftsfenster“, Werkbild BOCAD

6. Intelligente Hilfsfunktionen, Kollisionskontrolle, Überhöhung

Ein CAD-System der Hochleistungsklasse zeichnet sich u. a. durch bauspezifisch intelligente Hilfsfunktionen aus.

Eine interessante, zur Einführung in intelligente Hilfsfunktionen noch gut überschaubare Leistung ist die Berechnung und Darstellung des Schwerpunkts beliebiger Fertigungs- und Montageeinheiten. Der gemeinsame Schwerpunkt einer Liste von Elementen, z. B. einer Hauptposition mit Leitteil und Anbauteilen (Bild 6-1), kann gesucht sein, um dort ein Blech mit Transportauge oder eine Öse zu konstruieren.

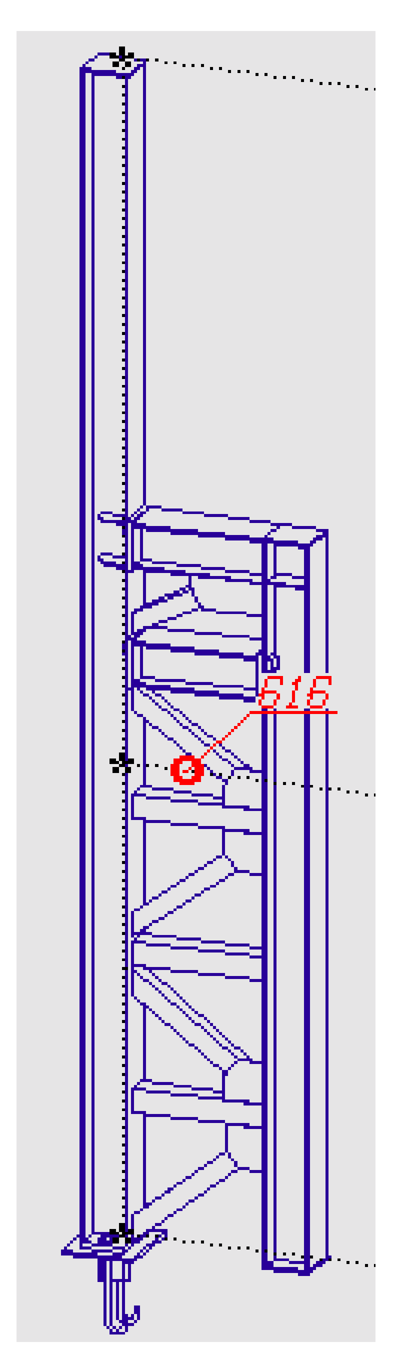


Bild 6-1: Schwerpunkt einer Hauptposition, automatisch ermittelt

Es erleichtert die Konstruktionsarbeit ungemein, wenn für diese Leistung in einem besonderen Fenster, das ausschließlich die gewählte Einheit darstellt, über das Symbol „C.G.“ (international „Center of Gravity“) die Berechnung automatisch erfolgt und der Schwerpunkt zur weiteren Nutzung als Punktobjekt dargestellt wird. Der Schwerpunkt wird in jedem offenen Fenster, in dem die Hauptposition enthalten ist, als globales Punktobjekt umrandet und mit einer Punktnummer angezeigt, siehe Punkt 616 in Bild 6-1 und 6-2.

Dabei werden die Daten des gemeinsamen Schwerpunktes aller Teile der gewählten Liste (in Bild 6-1 eine Hauptposition) unter Vernachlässigung von Schrauben- und Nahtgewicht errechnet. Die Teilegeometrie mit Ausklinkungen, Eckabschnitten und Durchbrüchen wird dabei

exakt ausgewertet. Auch der Schwerpunkt von gebogenen Teilen oder Verbundkonstruktionen mit unterschiedlichem spezifischen Gewicht wird korrekt ermittelt, siehe Bild 6-3.

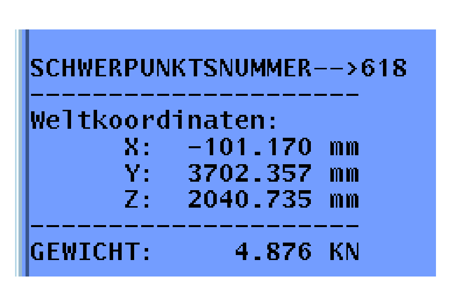


Bild 6-2: Schwerpunktdaten

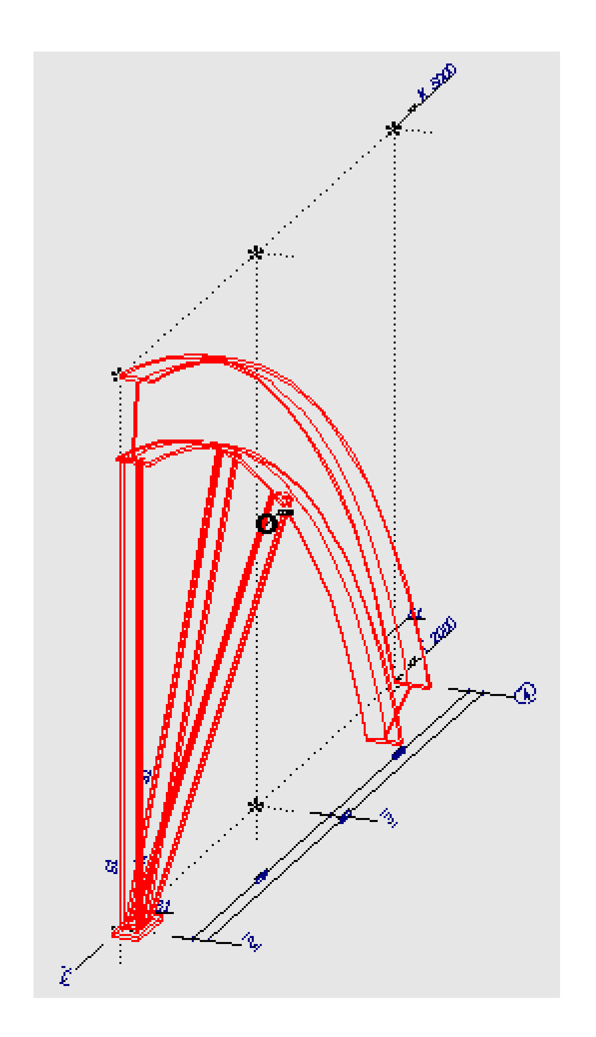


Bild 6-3: Schwerpunkt von gebogenen Teilen

Im Eigenschaftsfenster werden die Schwerpunktdaten mit Punkt-nummer ausgegeben, Koordinaten im 3D-Raum auf den Ursprung bezogen und das Gesamtgewicht aller gewählten Teile in KN berechnet.

Komplexer als die Hilfsfunktion „Schwerpunktermittlung“ gestaltet sich die Hilfsfunktion zur Herleitung von Überhöhungen für Träger und Fachwerke. Zwei über wenig erforschte, z. T. strittige Logik, miteinander zusammenhängende geometrische Zustände für dasselbe Baugruppenobjekt sind bei Überhöhungen zu beachten. Zum Einen ist dies der geometrische Zustand nach der Montage unter Durchbiegung durch die Belastung, für deren Kompensation die Überhöhung gedacht ist. Dieser Zustand wird üblicherweise in Genehmigungsplänen und Übersichten dargestellt. Zum Anderen ist dies der überhöhte Zustand während der Fertigung und dem Zusammenbau in der

Werkstatt. Für das gleiche Objekt sind also je nach Nutzung der daraus hergeleiteten Zeichnung zwei geometrisch verschiedene Zustände zu speichern und zu verwalten. Die Hilfsfunktion „Überhöhung“ soll dem CAD-Ingenieur maßgeblich helfen, aus der Geometrie ohne Überhöhung die Geometrie mit Überhöhung in wenigen Zügen herzuleiten.

Dabei sind mehrere, prinzipiell unterschiedliche Fälle zu unterscheiden. Der einfachste, banale Fall sind Walzprofilträger, die erst nach der Bearbeitung (Bohren, Körnen, Sägen) so wenig überhöht werden, dass kein Einfluss auf die Lage der Bohrungen, den Sägewinkel und die über Körnungen zu platzierenden Anbauteile zu berücksichtigen ist. Für diesen einfachen Fall genügt ein Werkstatt-Kommentar am Leitteil „Nach Bearbeitung x mm überhöhen“. Das Leitteil kann dann als gerader Träger auf einem automatischen NC-Bearbeitungszentrum fertig bearbeitet werden und wird anschließend oder gar erst nach Anschweißen der Anbauteile in der Werkstatt durch Wärmepunkte oder eine hydraulische Presse gebogen.

Bei stark überhöhten, insbesondere hohen Trägern muss die kinematische Bewegung der Bohrungen, Körnungen und der Sägewinkel durch Überhöhung berechnet werden, wenn erst nach der NC-Bearbeitung des geraden Trägers gebogen wird.

Nicht einig sind sich offenbar die Experten der verschiedenen Stahlbauunternehmen, ob dann die Sägewinkel an den Trägerenden für den belasteten Zustand oder den unbelasteten Zustand bei der Montage zu wählen sind.

Noch komplexer werden die Verhältnisse bei Fachwerkträgern, bei denen sich die Länge der Diagonalstäbe und Pfosten sowie die Winkel, unter denen sie an die Gurte angeschlossen werden, durch Überhöhung des Untergurts und ggf. auch des Obergurts ändern.

Die eleganteste und einfachste Hilfe für den CAD-Ingenieur am Bildschirm ist eine Hilfsfunktion „Überhöhung“ analog zum Durchmessergriff einer Linie, die über den Griff zum Kreisbogensegment verformt wird. Mit einem Durchmessergriff am Gurt ist so der zunächst gerade Gurt wunschgemäß zu überhöhen. Durch spezifische Intelligenz der CAD-Methode „Überhöhung" passen sich dann alle Diagonalstäbe

und ihre Anschlüsse an den Gurt durch Vererbung der Anschlussmethode der neuen, überhöhten Geometrie an. Diese wird dann als Werkstattzeichnung dokumentiert.

7. Ausblick und Zusammenfassung

Selbst in Ländern mit hervorragender Bauingenieurausbildung ist zu beobachten, dass sich in der Baupraxis nur leicht und effizient zu bedienende CAD-Systeme durchsetzen, deren erster Eindruck Laien und Fachleute überzeugt. Ein wesentliches Forschungsziel der Bauinformatik ist deshalb, die Bedienung durch intelligentes Verhalten von CAD-System selbst bei komplexen Aufgabenstellungen einfach zu gestalten. Die Nutzoberfläche von Programmen ist der entscheidende Schlüssel, ob Forschung zum Erfolg in der Praxis des Bauwesens führt. Für leistungsstarke Branchensoftware wird zudem globaler, weltweiter Einsatz typisch. Ideale Eigenschaft dazu wäre die Unabhängigkeit der Bausoftware von Landessprachen und Schriftzeichen. Diese Ziele scheinen erreichbar durch textfreie, selbsterklärende oder allgemein bekannte Anfasser an Objekten kombiniert mit bauspezifisch intelligentem Verhalten der Objekte. Interaktive Aktionen, z. B. Verlängern eines Stabes durch Ziehen am Anfasser seiner Stirnseite, werden dadurch außerordentlich einfach und direkt. Selbst für komplexe Aktionen hat der CAD-Ingenieur am Bildschirm lediglich anstelle vieler einzelner Lösungsschritte die gewünschte Aufgabe über Anfasser am intelligenten Objekt zu stellen. Jedes Objekt bietet für die mit ihm möglichen Aktionen, z. B. Verschweißen, Verschrauben, Verschieben, Kopieren, Löschen etc. aufgabentypische Anfasser an. Die entsprechende Lösung liefert dann das CAD-System automatisch durch intelligentes, branchentypisches Objektverhalten. Die Idee, auf direkte Aktionen mit Anfassern intelligent reagierende Objekte zu entwickeln, ist von einfachen Objekten des Bauwesens, wie Trägern und Blechen, bis hin zu hochkomplexen Objekten, wie kompletten Treppentürmen, kontinuierlich auszubauen. Die vorliegende Arbeit leistet einen Beitrag, diesem hochgesteckten Ziel nahe zu kommen und so die Baupraxis weltweit weiter zu bringen. Europäische Hochtechnologie auf diese Weise von Landessprachen unabhängig zu machen und im eigenen Heimatland China einzusetzen, ist ein besonderes Anliegen des Verfassers der Arbeit.

In dieser Arbeit werden daher systematisch Elemente einer Nutzoberfläche gesucht und entwickelt, die kontextsensitiv ein direktes Arbeiten am Objekt erlauben, also ohne verschachtelte Menüfelder und ohne umfangreiche Symbolanhäufungen an den Bildrändern auskommen. Mit diesen Elementen werden neue Vorgehensweisen

konzipiert, die unter Voraussetzung intelligenten Objektverhaltens effizienteres, eingabearmes Konstruieren ermöglichen. Was dabei „intelligentes Objektverhalten“ ist, wird konkret an ein-, zwei- und dreidimensionalen Objekten diskutiert und an ausgewählten Beispielen programmtechnisch verwirklicht. Auf diese Weise wird die entwickelte Theorie einer von Landessprachen unabhängigen Nutzoberfläche mit intelligenten CAD-Objekten im Experiment ingenieurwissenschaftlich nachgewiesen. Besonders anspruchsvoll ist dabei die Randbedingung, diese Leistung in ein lebendes CAD-System der oberen Leistungsklasse durch Re-Engineering störungsfrei zu integrieren. In weiterführenden Arbeiten können dann, aufbauend auf dem Erfahrungsfundus dieser Arbeit, immer komplexere Aktionen mit bautypischer Intelligenz automatisiert werden, so dass sich die Bedienung von CAD-Systemen für den Anwender immer einfacher und effizienter gestaltet.

Literatur

[1] Girmscheid, G.: „Neue unternehmerische Strategien in der Bauwirtschaft - Systemanbieterwettbewerb und virtuelle Unternehmen“.
Institut für Bauplanung und Baubetrieb ETH Zürich (1997)

[2] Caglayan, A.: Intelligente Software Agenten. München, Wien: Hanser 1998

[3] Klusch, M.: Intelligent Information Agents. Berlin, Heidelberg: Springer 1998

[4] Sayegh, A.: CORBA Standard, Spezifikationen, Entwicklung. Köln: O'Reilly 1999

[5] Bretschneider, D.: Modellierung rechnerunterstützter, kooperativer Arbeit in der Tragwerksplanung. Fortschr.-Ber. VDI Reihe 4 Nr. 151. Düsseldorf: VDI-Verlag 1998

[6] Grabowski, H., Rude, S., Lossak, R.-S.: verteilte Modellierung komplexer Produkte durch Einführung von Konstruktionsarbeitsräumen. VDI Berichte, 1289: 197-220, 1996

[7] Richard J. Gaylord, Samnel N. Kamin, Paul R. Wellin: Einführung in die Programmierung mit Mathematica; Basel, Birkhäuser Verlag, 1995
ISBN 3-7643-2965-3

[8] Balzert, H.: Lehrbuch der Software-Technik: Software Entwicklung
Heidelberg ; Berlin: Spektrum, Akad. Verl., 2000
ISBN 3-8274-0480-0

[9] René Martin: Visio 2000 professionell
München; Boston: Addison-Wesley, 2000
ISBN 3-8273-1626-X

[10] Ludorff, Kolja: CAD-Leitbilder, Konzeption und Entwicklung
Diplomarbeit am Lehr- und Forschungsgebiet Bauinformatik der Bergischen Universität - GH Wuppertal, Fachbereich Bauingenieurwesen (2000)

[11] Ignaz, Walter: Inhalte vor Zertifikaten
In: Forschung & Lehre, Heft 9/2000, Mitteilungen Deutscher Hochschulverband, www.forschung-und-lehre.de

[12] Hoppe, Christian: Vergleich direkter, textfreier CAD-Nutzoberflächen
Diplomarbeit am Lehrgebiet Bauinformatik der Bergischen Universität GH Wuppertal (2000)

[13] Hanff, Jochen; Kasparek, Eva; Ruess, Martin; Schutte, Gerrit (Hrsg.):
Forum Bauinformatik 2000-10-05
Fortschr.-Ber. VDI Reihe 4 Nr. 163. Düsseldorf: VDI Verlag 2000
ISBN 3-18-316304-3

[14] Oppermann, Johannes: Das Einsteigerseminar CorelDRAW 8
Kaarst: bhv Verlag 1998, ISBN 3-8287-1007-7

[15] Hoffmann, Thomas (Hrsg.): Corel Draw 9
München: Ullstein Taschenbuchverlag 2000, ISBN 3-548-41162-2

[16] Vogel, Harald: Das Einsteigerseminar AutoCAD 2000
Kaarst: bhv Verlag 2000, ISBN 3-8287-1071-9

[17] Balzert, Heide: Lehrbuch der Objektmodellierung: Analyse und Entwurf
Heidelberg; Berlin: Spektrum, Akad. Verl., 1999
ISBN 3-8274-0285-9

[18] Andreas Heuer: Objektorientierte Datenbanken - Konzepte, Modelle, Standards und Systeme; Addison-Wesley-Longman, 1997, ISBN 3-89319-800-8

[19] Günter Reinemann: CAD mit ACAD-Bau; Braunschweig; Wiesbaden: Vieweg, 1995, ISBN 3-528-06605-9

[20] Landscheidt, Willi: Bauzeichnungen; Ausburg: Augustus-Verl., 1996
ISBN 3-8043-0472-9

[21] Balzert, Helmut: Einführung in die Informationsverarbeitung; Ismaning: Hueber-Holzmann, 1989, ISBN 3-19-009856-5

[22] Balzert, Helmut: Konzepte und Notationen in UML, Java und C++, Algorithmik und Software-Technik, Anwendungen; Heidelberg: Spektrum, Akad. Verl., 1999, ISBN 3-8274-0358-8

[23] Scott Urman: Oracle 8i Advanced PL/SQL Programming; McGraw-Hill Companies; 2000, ISBN 0-07-212146-7

[24] Glaeser, Georg: Fast algorithms for 3D-graphics; New York: Springer-Verlag; 1994, ISBN 0-387-94288-2

[25] Oestereich, Bernd: Objektorientierte Softwareentwicklung: Analyse und Design mit der Unified modeling language;
Wien: Oldenbourg; 1998, ISBN 3-486-24787-5

[26] James D. Foley; Computer graphics: principles and practice; Addison-Wesley Publishing Company; 1990, ISBN 0-201-12110-7

[27] Steven Harrington: Computergraphics - Einführung durch Programmierung;
Hamburg, McGraw-Hill Book Company GmbH;
1998, ISBN 3-89028-122-2

[28] Richard Cooper: Objekt Databases: An ODMG Approach; London, International Thomson Computer Press; 1997, ISBN 1-85032-294-5

[29] R. G. G. Cattell: The Objekt Database Standard: OPMG 2.0; San Francisco: Morgan Kaufmann Publishers; 1997, ISBN 1-55860-463-4

[30] René Klösch: Objektorientiertes Reverse Engineering: von klassischer zu Objektorientierter Software; Berlin, Heidelberg, Spinger-Verlag; 1995, ISBN 3-540-58374-2

[31] Kevin Jameson: Multi-Platform Code Management; Sebastopol, O'Reilly & Associates; 1994, ISBN 1-56592-059-7

[32] Steve Maguire; Strategien der Software - Entwicklung; Unterschleißheim, Microsoft Press Deutschland; 1994, ISBN 3-86063-338-4

[33] Jeff Johnson: GUI Bloopers: Don'ts and Do's for Software Developers and Web Designers; Morgan Kaufmann Publishers; 2000, ISBN 1-55860-582-7

[34] Barth, R., Beier, E.: Grafikprogrammierung mit Open GL; Bonn, Addison-Wesley (Deutschland) GmbH; 1996, ISBN 3-89319-975-6

[35] Woo, Mason: OpenGL programming guide: the official guide to learing OpenGL; Addison-Wesley Developers Press; 1996, ISBN 0-201-46138-2

[36] Xie, Buyin; Lehrbuch für Computer Grafik; Shanghai, Tongji Universität Press; 1995, ISBN 7-5608-1566-9 (Chinesisch)

[37] Guo, Chaoyong: Technik der Entwicklung AutoCAD R14; Beijing, Tsinghua Universität Press; 1999, ISBN 7-302--03399-4 (Chinesisch)

[38] AutoCAD 2000 Benutzerhandbuch; Autodesk Development S.à.r.l., Nenchâtel, Schweiz, 1999

[39] Andreas Solymosi: Objektorientiertes Plug and Play: Ein Programmierlehrbuch für Wiederverwendbarkeit und Softwarequalität in C++; Wiesbaden: Vieweg, 1997, ISBN 3-528-05569-3

[40] AutoCAD 2000 bietet die Grundlage für leistungsfähige Applikationen; CAD-CAM REPORT: Engineering Magazin für C-Technologien; Heidelberg, Dressler Verlag, Nr.3 März 1999

[41] Guido Krueger: Go to Java 2; München: Addision-Wesley Verlag, 2000,
ISBN 3-8273-1710-X

[42] Liwu Li: Java Date Structures and Programming; Heidelberg: Springer Verlag, 1998, ISBN 3-540-63763-X